

带I²C总线/SPI接口, 64 字节发送和接收FIFO, 支持内置 IrDA(版本 1.0 和 1.1)的双UART—SC16IS752/SC16IS762

1.概述.....	3
2.特性.....	3
2.1 通用特性	3
2.2 I ² C总线特性.....	4
2.3 SPI特性	4
3.应用	4
4.订购信息.....	4
5.方框图.....	5
6.管脚配置信息.....	6
6.1 管脚配置	6
6.2 管脚描述	7
7.功能描述.....	9
7.1 触发点	9
7.2 硬件流控制	9
7.2.1 自动 $\overline{\text{RTS}}$	10
7.2.2 自动 $\overline{\text{CTS}}$	10
7.3 软件流控制	11
7.3.1 RX.....	12
7.3.2 TX	12
7.4 硬件复位, 上电复位(POR)和软件复位	13
7.5 中断	13
7.5.1 中断模式操作.....	14
7.5.2 查询模式操作.....	14
7.6 睡眠模式	15
7.7 间隔和超时条件	15
7.8 可编程波特率发生器	15
8.寄存器描述.....	18
8.1 接收保存寄存器(RHR)	20
8.2 发送保存寄存器(THR).....	20
8.3 FIFO控制寄存器(FCR)	21
8.4 线控制寄存器(LCR).....	21
8.5 线状态寄存器(LSR)	23
8.6 MODEM控制寄存器(MCR).....	24
8.7 MODEM状态寄存器(MSR)	24
8.8 中断使能寄存器(IER)	25

8.9 中断识别寄存器(IIR)	26
8.10 增强型特性寄存器(EFR)	27
8.11 除数寄存器(DLL, DLH)	27
8.12 发送控制寄存器(TCR)	27
8.13 触发点寄存器(TLR)	28
8.14 发送器FIFO电平寄存器(TXLVL)	28
8.15 接收器FIFO电平寄存器(RXLVL)	28
8.16 可编程的I/O脚方向寄存器 (IODIR)	28
8.17 可编程的I/O脚状态寄存器(IOSSTATE)	29
8.18 I/O中断使能寄存器(IOINTENA)	29
8.19 I/O控制寄存器 (IOCONTROL)	29
8.20 额外特性控制寄存器(EFCR)	30
9.RS-485 特性.....	31
9.1 自动RS-485 $\overline{\text{RTS}}$ 控制	31
9.2 RS-485 $\overline{\text{RTS}}$ 输出翻转	31
9.3 自动RS-485	31
9.3.1 正常多点模式	31
9.3.2 自动地址检测	32
10.I²C总线操作	32
10.1 数据传输	32
10.2 寻址和传输格式	34
10.3 寻址	35
10.4 子地址的使用	36
11.SPI操作.....	37
12.极限值.....	38
13.静态特性.....	38
14.动态特性.....	40
15.表面封装.....	46

1.概述

SC16IS752/SC16IS762 是I²C总线/SPI总线接口, 双通道高性能的UART提供高达 5Mbit/s 的数据率, 低操作和睡眠电流; 它还应用提供 8 个额外可编程的I/O脚。器件含有极小的 HVQFN32 和TSSOP28 封装, 使其理想适用于便携式和电池操作的应用中。

SC16IS752 在功能和电气特性上与 SC16IS762 相同, 不同的是 SC16IS752 所支持的 IrDA 速率高达 115.2kbit/s, 而 SC16IS762 所支持的 IrDA 速率高达 1.152Mbit/s。

SC16IS752/SC16IS762 的内部寄存器集向后兼容广泛使用和普遍流行的 16C450。这就使得软件可以容易编写或从另一个平台移植过来。

SC16IS752/SC16IS762 还提供其它高级的特性, 例如自动硬件和软件流控制, 自动的 RS-485 支持和软件复位。这允许软件可在任何时候复位 UART, 与硬件的复位信号无关。

2.特性

2.1 通用特性

- 双全双工 UART
- 可选的I²C总线或SPI接口
- 3.3V 或 2.5V 操作
- 工业级温度范围: -40℃~+85℃
- 64 字节 FIFO (发送器和接收器)
- 与工业标准 16C450 完全兼容并等效
- 在 16×时钟模式下波特率高达 5Mbit/s
- 使用 $\overline{\text{RTS}}/\overline{\text{CTS}}$ 的自动硬件流控制
- 带有可编程 Xon/Xoff 字符的自动软件流控制
- 一个或两个 Xon/Xoff 字符
- 自动的 RS-485 支持 (自动的从地址检测)
- 多达 8 个可编程的 I/O 脚
- 经过 $\overline{\text{RTS}}$ 信号的 RS-485 驱动器方向控制
- RS-485 驱动器方向控制翻转
- 内置 IrDA 编码器和译码器接口
- SC16IS752 支持的 IrDA 速率高达 115.2kbit/s
- SC16IS762 支持的 IrDA 速率高达 1.152Mbit/s。
- 软件复位
- 发送器和接收器可相互独立使能/禁能
- 接收和发送 FIFO 电平
- 可编程的特殊字符检测
- 完全可编程的字符格式:
 - ◆ 5, 6, 7 或 8 位字符
 - ◆ 偶、奇或无奇偶格式
 - ◆ 1, 1 $\frac{1}{2}$ 或 2 个停止位

- Line break 的产生和检测
- 内部回送模式
- 3.3V 时的睡眠电流低于 30 μ A
- 工业和商业温度范围
- 5V 容限输入
- 可用于 HVQFN32 和 TSSOP28 封装

2.2 I²C 总线特性

- SCL/SDA 输入上的噪声滤波器
- 400kbit/s (最大速率)
- 遵循 I²C 总线高速
- 仅为从机模式

2.3 SPI 特性

- 最高速率为 4Mbit/s
- 仅为从机模式
- SPI 模式 0

3.应用

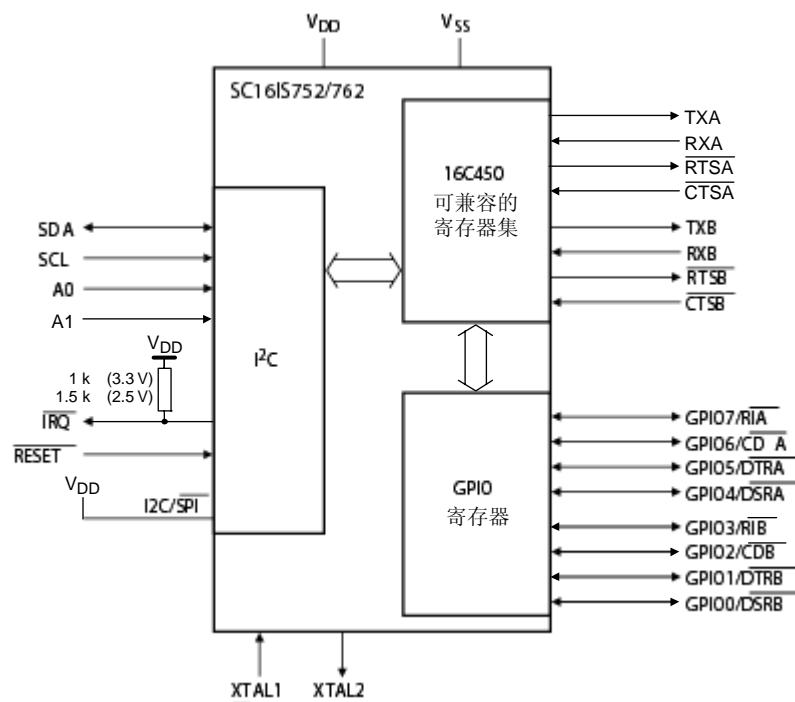
- 制造自动化和进程控制
- 便携式和电池操作的设备
- 单元数据设备

4.订购信息

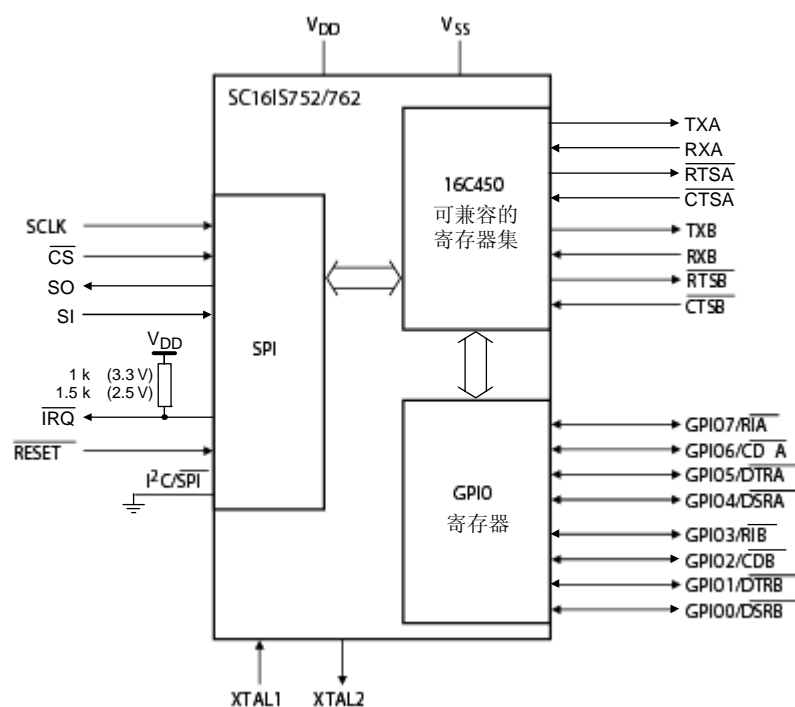
表 1 订购信息

器件型号	封装		
	名称	描述	版本
SC16IS752IPW	TSSOP28	TSSOP 封装; 28 脚; 本体宽度 4.4mm	SOT361-1
SC16IS762IPW	TSSOP28	TSSOP 封装; 28 脚; 本体宽度 4.4mm	SOT361-1
SC16IS752IBS	HVQFN32	HVQFN 封装; 无引脚; 32 端	SOT617-3
SC16IS762IBS	HVQFN32	HVQFN 封装; 无引脚; 32 端	SOT617-3

5.方框图



a. I²C- 总线接口



b. SPI接口

图 1 SC16IS752/SC16IS762 的方框图

6. 管脚配置信息

6.1 管脚配置

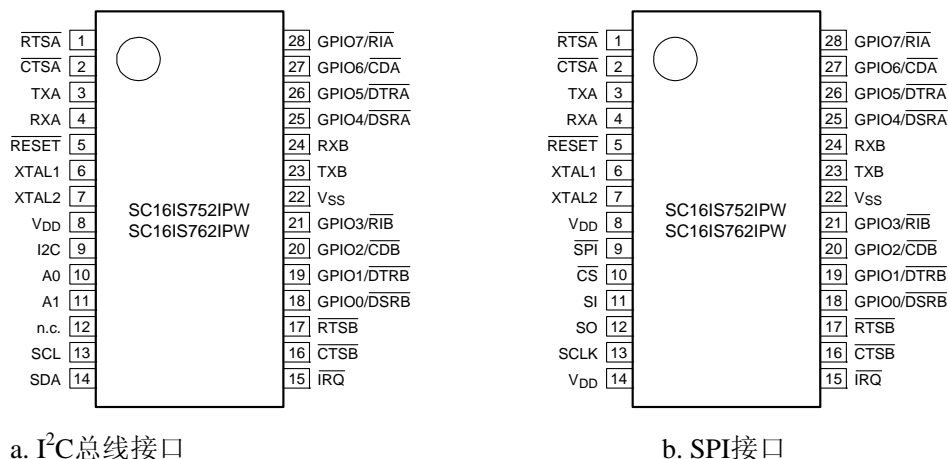


图 2 TSSOP28 的管脚配置

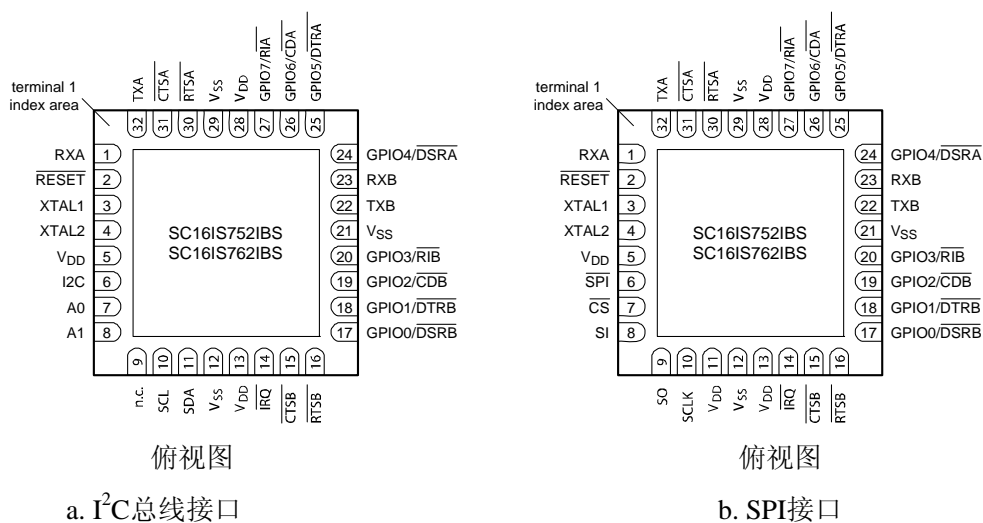


图 3 HVQFN32 的管脚配置

6.2 管脚描述

表 2 管脚描述

符号	管脚		类型	描述
	TSSOP28	HVQFN32		
$\overline{\text{CS}}/\text{A0}$	10	7	I	SPI芯片选择或I ² C总线器件地址选择A0。如果SPI配置由I2C/ $\overline{\text{SPI}}$ 管脚选择,那么该管脚为SPI芯片选择管脚(施密特触发,低电平有效)。如果I ² C总线配置由I2C/ $\overline{\text{SPI}}$ 管脚选择,那么用户可通过该管脚和A1脚来改变器件的基址。如需选择器件地址,请参考表 32。
$\overline{\text{CTSA}}$	2	31	I	UART 清除发送(低电平有效)。 $\overline{\text{CTSA}}$ 管脚上的逻辑0(低电平)表示 modem 或数据集准备接受 SC16IS752/SC16IS762 的发送数据。可通过读 MSR [4]测试状态。当通过增强型特性寄存器 EFR[7]使能自动 $\overline{\text{CTS}}$ 功能用于硬件流控制操作时,该管脚仅影响发送和接收操作。
$\overline{\text{CTSB}}$	16	15	I	UART 清除发送(低电平有效),通道 B。 $\overline{\text{CTSB}}$ 管脚上的逻辑0表示 modem 或数据集准备接受 SC16IS752/SC16IS762 的发送数据。可通过读 MSR [4]测试状态。当通过增强型特性寄存器 EFR[7]使能自动 $\overline{\text{CTS}}$ 功能用于硬件流控制操作时,该管脚仅影响发送和接收操作。
I2C/ $\overline{\text{SPI}}$	9	6	I	I ² C总线或SPI接口选择。如果该管脚为高电平,则选择I ² C总线接口。如果该管脚为低电平,则选择SPI接口。
$\overline{\text{IRQ}}$	15	14	O	中断(开漏、低电平有效)。当中断源在中断使能寄存器(IER)中使能时,中断被使能。中断条件包括:输入管脚的状态变化,接收错误,可用的接收缓冲数据,可用的发送缓冲空间,或当检测到 modem状态标志时。在该管脚和V _{DD} 之间必须连接一个外部电阻(3.3V时 1K Ω , 2.5V时 1.5K Ω)。
SI/A1	11	8	I	SPI数据输入管脚或I ² C总线器件地址选择A1。如果SPI配置由I2C/ $\overline{\text{SPI}}$ 管脚选择,则该管脚为SPI数据输入管脚。如果I ² C总线配置由I2C/ $\overline{\text{SPI}}$ 管脚选择,那么用户可通过该管脚和A0脚来改变从机的基址。如需选择器件地址,请参考表 32。
SO	12	9	O	SPI数据输出管脚。如果SPI配置由I2C/ $\overline{\text{SPI}}$ 管脚选择,那么该管脚为可三态输出的管脚。如果I ² C总线配置由I2C/ $\overline{\text{SPI}}$ 管脚选择,那么该管脚未定义且必须悬空。
SCL/SCLK	13	10	I	I ² C总线或SPI输入时钟。

续上表

符号	管脚		类型	描述
	TSSOP28	HVQFN32		
SDA	14	11	I/O	如果I ² C总线配置由I2C/SPI管脚选择, 那么I ² C总线数据输入/输出为开漏模式。如果选择SPI配置, 该管脚不使用且必须连接到V _{DD} 。
GPI00/ $\overline{\text{DSRB}}$	18	17	I/O	可编程的I/O脚或modem $\overline{\text{DSRB}}$ ^[1]
GPI01/ $\overline{\text{DTRB}}$	19	18	I/O	可编程的I/O脚或modem $\overline{\text{DTRB}}$ ^[1]
GPI02/ $\overline{\text{CDB}}$	20	19	I/O	可编程的I/O脚或modem $\overline{\text{CDB}}$ ^[1]
GPI03/ $\overline{\text{RIB}}$	21	20	I/O	可编程的I/O脚或modem $\overline{\text{RIB}}$ ^[1]
GPI04/ $\overline{\text{DSRA}}$	25	24	I/O	可编程的I/O脚或modem $\overline{\text{DSRA}}$ ^[2]
GPI05/ $\overline{\text{DTRA}}$	26	25	I/O	可编程的I/O脚或modem $\overline{\text{DTRA}}$ ^[2]
GPI06/ $\overline{\text{CDA}}$	27	26	I/O	可编程的I/O脚或modem $\overline{\text{CDA}}$ ^[2]
GPI07/ $\overline{\text{RIA}}$	28	27	I/O	可编程的I/O脚或modem $\overline{\text{RIA}}$ ^[2]
$\overline{\text{RESET}}$	5	2	I	硬件复位（低电平有效） ^[3]
$\overline{\text{RTSA}}$	1	30	O	UART 请求发送（低电平有效），通道 A。 $\overline{\text{RTSA}}$ 管脚上的逻辑 0 表示发送器的数据就绪并等待发送。在 modem 控制寄存器 MCR[1]中写 1 将设置该管脚为 0，表示数据可用。复位后该管脚设置为逻辑 1。当自动 $\overline{\text{RTS}}$ 功能通过增强型特性寄存器 EFR[6]使能用于硬件流控制操作时，该管脚仅影响发送和接收操作。
$\overline{\text{RTSB}}$	17	16	O	UART 请求发送（低电平有效），通道 B。 $\overline{\text{RTSB}}$ 管脚上的逻辑 0 表示发送器的数据就绪并等待发送。在 modem 控制寄存器 MCR[1]中写 1 将设置该管脚为 0，表示数据可用。复位后该管脚设置为逻辑 1。当自动 $\overline{\text{RTS}}$ 功能通过增强型特性寄存器 EFR[6]使能用于硬件流控制操作时，该管脚仅影响发送和接收操作。
RXA	4	1	I	通道 A 接收器输入。在局部环回模式中，RXA 输入管脚被禁能且 TXA 数据内部连接到 UART RXA 输入。
RXB	24	23	I	通道 B 接收器输入。在局部环回模式中，RXB 输入管脚被禁能且 TXB 数据内部连接到 UART RXB 输入。
TXA	3	32	O	通道 A 发送器输出。在局部环回模式下，TXA 输出管脚被禁能且 TXA 数据内部连接到 UART RXA 输入。
TXB	23	22	O	通道 B 发送器输出。在局部环回模式下，TXB 输出管脚被禁能且 TXB 数据内部连接到 UART RXB 输入。
V _{DD}	8	5,13,28	-	电源电压。
V _{SS}	22	12,21,29	-	地。

续上表

符号	管脚		类型	描述
	TSSOP28	HVQFN32		
V _{SS}	-	Center pad	-	HVQFN32 封装背面的 center pad 为金属性并且在 PCB 板上应被连接到地。
XTAL1	6	3	I	晶体输入或外部时钟输入。在 XTAL1 和 XTAL2 之间连接一个晶体来形成内部振荡器电路(见图 11)。或者, 也可以将外部时钟连接到该管脚。
XTAL2	7	4	O	晶体输出。(同见 XTAL1)。XTAL2 用作晶体振荡器输出。

[1] 用 IOControl 寄存器位 2 来选择。

[2] 用 IOControl 寄存器位 1 来选择。

[3] 见 7.4 节“硬件复位, 上电复位(POR)和软件复位”。

7.功能描述

UART将执行外围器件或modem接收的数据字符的串行到I²C转换, 以及在主机发送的数据字符上执行I²C到串行的转换。SC16IS752/SC16IS762 UART的完整状态在功能操作期间可以由主机随时读取。

SC16IS752/SC16IS762 可设置为交替模式(FIFO 模式), 通过缓冲接收/发送的字符降低主机过多的软件开销。接收和发送 FIFO 都可存储多达 64 字符(包括接收器 FIFO 的每字符 3 个附加错误状态位)并且具有可选择或可编程的触发点。

SC16IS752/SC16IS762 包含可选择的硬件流控制和软件流控制。硬件流控制大大减少了软件开销且通过使用 RTS 输出和 CTS 输入信号自动控制串行数据流来提高系统效率。软件流控制通过使用可编程的 Xon/Xoff 字符自动控制数据流。

UART包括一个可编程波特率发生器, 它可通过一个在 1 和 (2¹⁶-1) 之间的分频器来分频时序基准时钟输入。

7.1 触发点

SC16IS752/SC16IS762 为接收器和发送器中断的产生提供各自可选择和可编程的触发点。复位后, 发送和接收 FIFO 都禁能, 因此实际上触发点是一个字符的默认值。通过 FCR 得到可选择的触发点。通过 TLR 得到可编程的触发点。如果 TLR 位被清零, 那么使用 FCR 中可选择的触发点。如果 TLR 位没有被清零, 那么使用 TLR 中可编程的触发点。

7.2 硬件流控制

硬件流控制包括自动 CTS 和自动 RTS (见图 4)。编程EFR[7:6]可以独立使能/禁能自动 CTS 和自动 RTS。

若使用自动 CTS, CTS 在 UART 发送数据之前必须有效。

自动 RTS 仅当在 FIFO 中有足够空间接收数据时使 RTS 输出有效, 而在 RX FIFO 满时使

$\overline{\text{RTS}}$ 输出无效。中止和恢复 TCR 中的触发点决定 $\overline{\text{RTS}}$ 有效/无效时的值。如果 TCR 位被清零，那么使用 FCR 中可选择的触发点来代替 TCR。

如果自动 $\overline{\text{CTS}}$ 和自动 $\overline{\text{RTS}}$ 都被使能，那么当 $\overline{\text{RTS}}$ 连接 $\overline{\text{CTS}}$ 时，除非接收 FIFO 中有足够的空间，否则不发生数据发送。因此，在硬件流控制过程中可消除超时错误。如果没有使能，当发送数据率超过接收 FIFO 服务等待时间时，将产生超时错误。

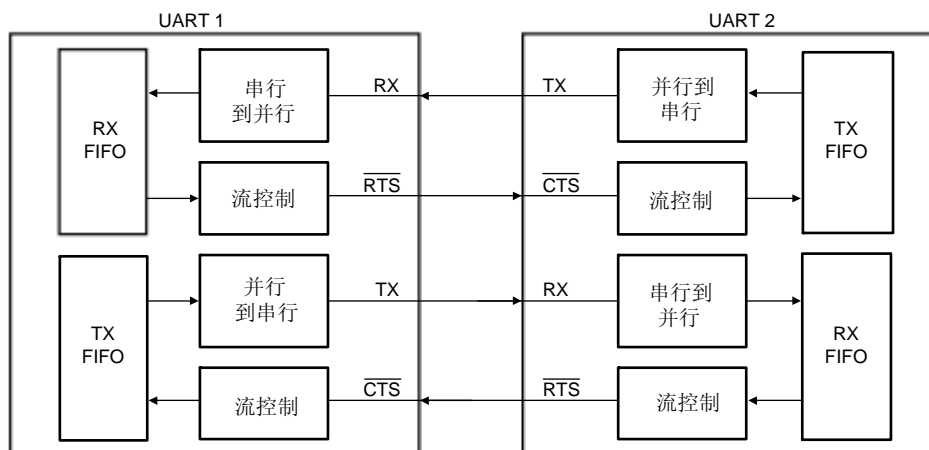
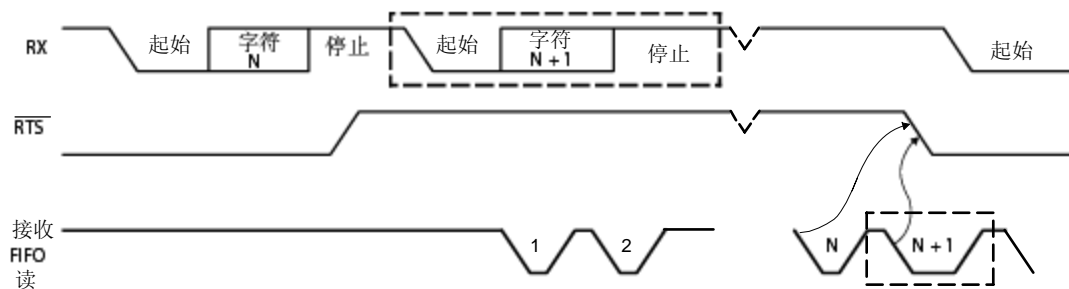


图4 自动流控制（自动 $\overline{\text{RTS}}$ 和自动 $\overline{\text{CTS}}$ ）举例

7.2.1 自动 $\overline{\text{RTS}}$

图5所示为 $\overline{\text{RTS}}$ 功能时序。自动 $\overline{\text{RTS}}$ 使用的接收 FIFO 触发点存储在 TCR 中。如果 RX FIFO 电平低于 TCR[3:0]内的中止触发点，则 $\overline{\text{RTS}}$ 有效。当到达接收 FIFO 的中止触发点时， $\overline{\text{RTS}}$ 无效。发送器件（如另一个 UART）可在到达触发点后发送一个另外的字符（假定发送 UART 有另一个字符要发送），因为发送器件将不能识别 $\overline{\text{RTS}}$ 的无效直至它开始发送其它的字符。一旦接收 FIFO 到达 TCR[7:4]编程的恢复触发点， $\overline{\text{RTS}}$ 就自动重新有效。重新有效使发送器件恢复发送。



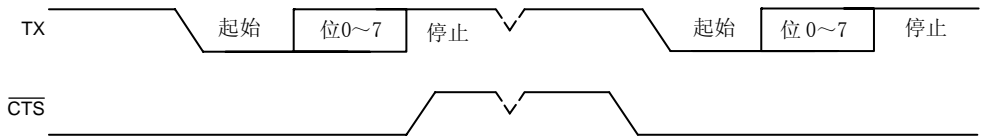
- (1) N= 接收 FIFO 触发点
- (2) 2个虚线框包含了有另外的字符被发送的情况，如7.2.1节所述。

图5 $\overline{\text{RTS}}$ 功能时序

7.2.2 自动 $\overline{\text{CTS}}$

图6所示为 $\overline{\text{CTS}}$ 功能时序。发送器电路在发送下一个数据字符之前检测 $\overline{\text{CTS}}$ 。当 $\overline{\text{CTS}}$ 有效时，发送器发送下一个字符。为了使发送器停止发送后面的字符， $\overline{\text{CTS}}$ 必须在当前发送的最后一个停止位的中间时刻之前被释放。自动 $\overline{\text{CTS}}$ 功能减少了向主机系统发送的中断。当流

控制被使能时,由于器件会自动控制各自的发送器,因此 $\overline{\text{CTS}}$ 电平的改变不会触发主机中断。若没有自动 $\overline{\text{CTS}}$,发送器发送出现在发送FIFO的任何数据并将导致接收器超时错误。



- (1) 当 $\overline{\text{CTS}}$ 为低时,发送器持续发送串行数据。
- (2) 如果 $\overline{\text{CTS}}$ 在当前字符的最后一个停止位的中间时刻之前变高,发送器完成当前字符的发送,但不发送下一个字符。
- (3) 当 $\overline{\text{CTS}}$ 从高变为低时,发送器重新开始发送数据。

图 6 $\overline{\text{CTS}}$ 功能时序

7.3 软件流控制

软件流控制通过增强型特性寄存器和modem控制寄存器使能。软件流控制的不同组合可通过设定EFR[3:0]的不同组合来使能。表 3所示为软件流控制选项。

表 3 软件流控制选项(EFR[3:0])

EFR[3]	EFR[2]	EFR[1]	EFR[0]	TX, RX 软件流控制
0	0	X	X	无发送流控制
1	0	X	X	发送 Xon1, Xoff1
0	1	X	X	发送 Xon2, Xoff2
1	1	X	X	发送 Xon1 和 Xon2, Xoff1 和 Xoff2
X	X	0	0	无接收流控制
X	X	1	0	接收器比较 Xon1, Xoff1
X	X	0	1	接收器比较 Xon2, Xoff2
1	0	1	1	发送 Xon1, Xoff1; 接收器比较 Xon1 或 Xon2, Xoff1 或 Xoff2
0	1	1	1	发送 Xon2, Xoff2; 接收器比较 Xon1 或 Xon2, Xoff1 或 Xoff2
1	1	1	1	发送 Xon1 和 Xon2, Xoff1 和 Xoff2 接收器比较 Xon1 和 Xon2, Xoff1 和 Xoff2
0	0	1	1	无发送流控制 接收器比较 Xon1 和 Xon2, Xoff1 和 Xoff2

还有两个与软件流控制有关的其它增强型特性:

- **Xon Any 功能(MCR[5]):** 在识别 Xoff 字符后接收任何字符将恢复操作。有可能 Xon1 字符被识别为 Xon Any 字符,这样就会使 Xon2 字符写入 RX FIFO。
- **特殊字符(EFR[5]):** 将输入的数据与 Xoff2 比较。特殊字符的检测设置 Xoff 中断(IIR[4])但并不中止发送。通过读 IIR 清除 Xoff 中断。特殊字符被传输到 RX FIFO。

7.3.1 RX

当软件流控制被使能时，SC16IS752/SC16IS762 将接收到的数据与 Xoff1/Xoff2 编程的字符相比较（在特定情况下，必须连续接收 Xoff1 和 Xoff2）。当接收到正确的 Xoff 字符时，传输在当前字符完全发送完后中止。Xoff 检测也设置 IIR[4]（如果通过 IER[5]使能）并使 \overline{IRQ} 变低。

要恢复发送，必须接收到 Xon1/Xon2 字符（在特定情况下，必须连续接收 Xon1 和 Xon2）。当接收到正确的 Xon 字符时，IIR[4]被清除且 Xoff 中断消失。

7.3.2 TX

当 RX FIFO 超过了 TCR[3:0]中设定的 HALT 触发点或 FCR[7:6]中可选择的触发点时，发送 Xoff1/Xoff2 字符。

当 RX FIFO 到达 TCR[7:4]中设定的 RESUME 触发点或下降到 FCR[7:6]中较低可选择触发点以下时，发送 Xon1/Xon2 字符。

Xoff/Xon 的发送和 FIFO 普通字符的发送的协议相同。这意味着尽管字长度设置为 5,6,7 位，也将发送 Xoff1/Xoff2, Xon1/Xon2 的最低位 5,6 或 7。（注意很少完成字符 5,6,7 位的发送，但所含的这个功能可保留与先前设计的兼容性。）

假定软件流控制和硬件流控制将永远不会被同时使能。图 7 所示为软件流控制的举例。

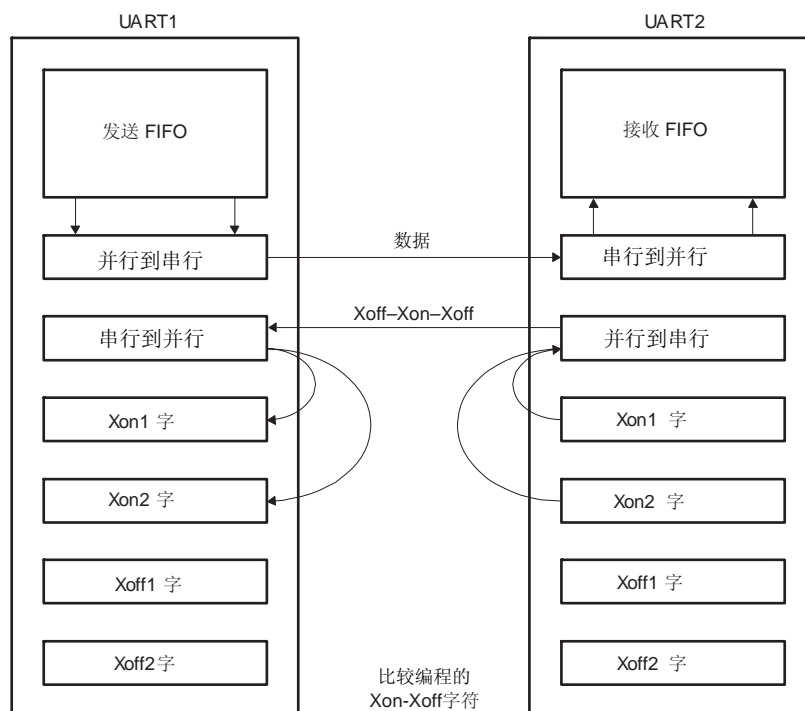


图 7 软件流控制的举例

7.4 硬件复位，上电复位(POR)和软件复位

这三种复位方式都相同并将复位表 4 中给出的内部寄存器。

表 4 总结了复位后寄存器的状态。

表 4 寄存器复位

寄存器	复位状态
中断使能寄存器	所有位清零。
中断识别寄存器	位 0 置位。其它所有位清零。
FIFO 控制寄存器	所有位清零。
线控制寄存器	复位到 00011101 (1Dh)。
Modem 控制寄存器	所有位清零。
线状态寄存器	置位位 5 和位 6。其它所有位清零。
Modem 状态寄存器	位 3:0 清零。位 7:4 输入信号。
增强型特性寄存器	所有位清零。
接收器保存寄存器	清除指针逻辑。
发送器保存寄存器	清除指针逻辑。
发送控制寄存器	所有位清零。
触发点寄存器	所有位清零。
发送 FIFO 电平	复位到 01000000 (40h)
接收 FIFO 电平	所有位清零。
I/O 方向	所有位清零。
I/O 中断使能	所有位清零。
I/O 控制	所有位清零。
额外特性寄存器	所有位清零。

- [1] 寄存器 DLL, DLH, SPR, Xon1, Xon2, Xoff1, Xoff2 都不是通过最高电平复位信号 $\overline{\text{RESET}}$, POR 和软件复位来进行复位，那就是说，它们在复位过程中保存它们的初始值。

表 5 总结了复位后输出信号的状态。

表 5 复位后的输出信号

信号	复位状态
TX	高
$\overline{\text{RTS}}$	高
I/O	输入
$\overline{\text{IRQ}}$	通过外部上拉变为高

7.5 中断

SC16IS752/SC16IS762 含有产生中断和优先级的功能 (7 种中断优先级)。中断使能寄存器(IER和IOIntEna)使能中断的每种类型和 $\overline{\text{IRQ}}$ 信号来响应中断的产生。当中断产生时，IIR 表示有中断等待处理并且在IIR[5:0]中提供中断的类型。表 6 总结了中断控制功能。

表 6 中断控制功能的总结

IIR[5:0]	优先级	中断类型	中断源
000001	无	无	无
000110	1	接收器线状态	RX FIFO 的字符中出现 OE, FE, PE 或 BI 错误
001100	2	RX 超时	RX FIFO 中的旧数据
000100	2	RHR 中断	接收数据就绪(FIFO 禁能)或 RX FIFO 在触发点以上 (FIFO 使能)
000010	3	THR 中断	发送 FIFO 空(FIFO 禁能)或 TX FIFO 经过触发点以上 (FIFO 使能)
000000	4	Modem 状态	Modem 输入管脚状态的改变
001110	5	I/O 脚	输入管脚状态改变
010000	6	Xoff 中断	接收 Xoff 字符/特殊字符
100000	7	$\overline{\text{CTS}}$, RTS	$\overline{\text{RTS}}$ 管脚或 $\overline{\text{CTS}}$ 管脚状态的改变从有效 (低电平) 到无效 (高电平)。

需要注意的是对于帧错误, 奇偶错误和间隔条件, LSR[7]都会产生中断。当在 RX FIFO 中的任何地方出现错误时 LSR[7]置位, 且仅当在 FIFO 中不再有剩余的错误时 LSR[7]清零。LSR[4:2]总是表示在 RX FIFO 顶部的接收字符的错误状态。读 RX FIFO 更新 LSR[4:2]为 FIFO 顶部新字符的适当状态。如果 RX FIFO 为空, 那么 LSR[4:2]全为 0。

对于 Xoff 中断, 如果 Xoff 流字符检测产生中断, 那么中断通过 Xon 流字符检测清除。如果特殊字符检测产生中断, 那么中断通过读 IIR 清除。

7.5.1 中断模式操作

在中断模式中 (如果 IER[3:0]的任意位为 1), 则通过中断信号 $\overline{\text{IRQ}}$ 告知主机接收器和发送器的状态。因此, 无需继续查询线状态寄存器 (LSR) 来判断是否还有中断需要被服务。图 8 所示为中断模式操作。

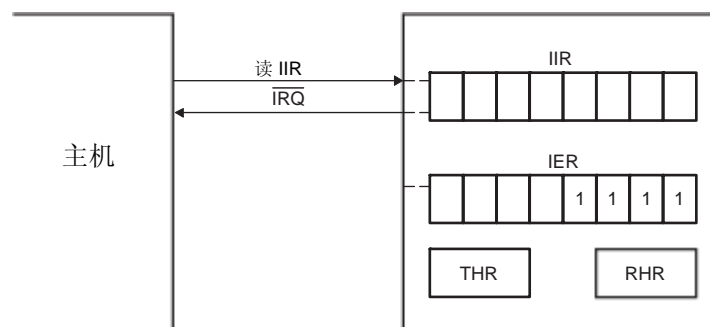


图 8 中断模式操作

7.5.2 查询模式操作

在查询模式中 (IER[3:0]=0000), 可通过查询线状态寄存器 (LSR) 检查接收器和发送器的状态。该模式是 FIFO 中断模式操作的一种选择, 在 FIFO 中断模式操作中, 通过中断发送到 CPU 的方式自动识别接收器和发送器的状态。图 9 所示为 FIFO 查询模式操作。

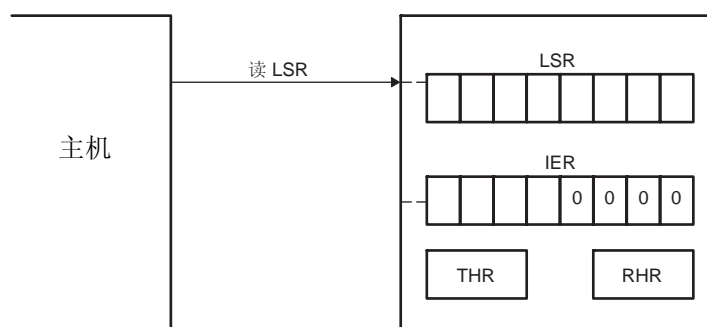


图 9 FIFO 查询模式操作

7.6 睡眠模式

睡眠模式是 SC16IS752/SC16IS762 UART 的增强型特性。当增强型功能位(EFR[4])置位且 IER[4]置位时睡眠模式使能。当出现下面几种情况时进入睡眠模式：

- 串行数据输入线RX为空闲（见7.7节“间隔和超时条件”）
- TX FIFO 和 TX 移位寄存器为空
- 没有中断等待处理，THR 除外

注：如果 RX FIFO 中有数据，那么器件将不会进入睡眠模式。

在睡眠模式下，UART 的时钟停止。由于大部分寄存器都使用这些时钟计时，因此功耗大大降低。当在 RX 线上检测到任何变化，在 modem 输入管脚的状态发生任何变化，或有数据写入 TX FIFO 时，UART 将唤醒。

注：在睡眠模式期间，必须不能写除数锁存器 DLL 和 DLH 来设置波特率时钟。因此，建议在写 DLL 或 DLH 之前使用 IER[4]来禁止睡眠模式。

7.7 间隔和超时条件

当 UART 接收到大量字符且这些数据不足以触发接收中断（因为它们没有到达接收触发点）时，UART 将在接收到最后一个字符后 4 字符时间内产生超时中断。超时计数器将在接收到的每个停止位的中间或每次读接收 FIFO 时复位。

当 RX 管脚被拉低的持续时间长于它发送一个完整字符加上起始(START)、停止(STOP)和奇偶(Parity)位所需要的时间时，检测到一个间隔条件。可通过置位 LCR[6]来发送间隔条件，出现这种情况时 TX 管脚将被拉低直至软件清零 LSR[6]。

7.8 可编程波特率发生器

SC16IS752/SC16IS762 UART 含有一个可编程波特率发生器，该波特率发生器采用任何时钟输入且通过分频器在 1 到 $(2^{16}-1)$ 之间的范围内将其分频。也可得到一个额外的 4 分频-预分频器以及由 MCR[7]选择，如图 10 所示。波特率发生器的输出频率为 $16 \times$ 波特率。除数的公式为：

$$\text{除数} = \frac{\left(\frac{\text{XTAL1晶体输入频率}}{\text{预分频}} \right)}{\text{所需的波特率} \times 16}$$

其中：

预分频=1，当复位后 MCR[7]被设为 ‘0’ 时（选择 1 分频-时钟）

预分频=4，当复位后 MCR[7]被设为 ‘1’ 时（选择 4 分频-时钟）

注：复位后预分频的默认值为 1 分频。

图 10显示内部预分频器和波特率发生器电路。

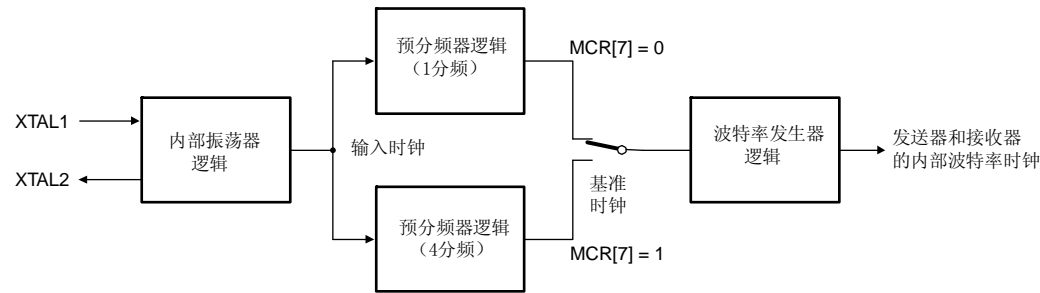


图 10 预分频器和波特率发生器方框图

必须写入 DLL 和 DLH 以便于编程波特率。DLL 和 DLH 分别是波特率分频值的最低和最高位字节。如果 DLL 和 DLH 都为 0，则 UART 被有效禁能，波特率时钟将不会产生。

备注：提供可编程的波特率发生器来选择发送和接收时钟率。

表 7和表 8分别列出使用 1.8432MHz和 3.072MHz晶体的波特率和除数值。

图 11所示为晶体时钟电路参考。

表 7 使用 1.8432MHz 晶体的波特率

所需波特率 (bit/s)	产生 16×时钟的除数值	所需波特率和实际波特率之间的百分比误差
50	2304	0
75	1536	0
110	1047	0.026
134.5	857	0.058
150	768	0
300	384	0
600	192	0
1200	96	0
1800	64	0
2000	58	0.69
2400	48	0
3600	32	0
4800	24	0
7200	16	0
9600	12	0
19200	6	0

续上表

所需波特率 (bit/s)	产生 16×时钟的除数值	所需波特率和实际波特率之间的百分比误差
38400	3	0
56000	2	2.86

表 8 使用 3.072MHz 晶体的波特率

所需波特率 (bit/s)	产生 16×时钟的除数值	所需波特率和实际波特率之间的百分比误差
50	2304	0
75	2560	0
110	1745	0.026
134.5	1428	0.034
150	1280	0
300	640	0
600	320	0
1200	160	0
1800	107	0.312
2000	96	0
2400	80	0
3600	53	0.628
4800	40	0
7200	27	1.23
9600	20	0
19200	10	0
38400	5	0

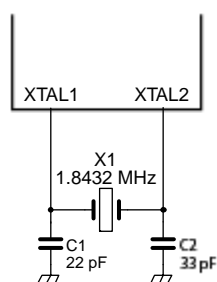


图 11 晶体振荡器电路参考

8.寄存器描述

表 9给出的是寄存器选择的编程组合。

表 9 寄存器映射—读/写特性

寄存器名称	读模式	写模式
RHR/THR	接收保存寄存器(RHR)	发送保存寄存器(THR)
IER	中断使能寄存器(IER)	中断使能寄存器
IIR/FCR	中断识别寄存器(IIR)	FIFO 控制寄存器(FCR)
LCR	线控制寄存器(LCR)	线控制寄存器
MCR	Modem控制寄存器(MCR) ^[1]	Modem控制寄存器 ^[1]
LSR	线状态寄存器(LSR)	n/a
MSR	Modem 状态寄存器(MSR)	n/a
SPR	暂存寄存器(SPR)	暂存寄存器
TCR	发送控制寄存器(TCR) ^[2]	发送控制寄存器 ^[2]
TLR	触发点寄存器(TLR) ^[2]	触发点寄存器 ^[2]
TXLVL	发送 FIFO 电平寄存器	n/a
RXLVL	接收 FIFO 电平寄存器	n/a
IODir	I/O 脚方向寄存器	I/O 脚方向寄存器
IOState	I/O 脚状态寄存器	n/a
IOIntEna	I/O 中断使能寄存器	中断使能寄存器
IOControl	I/O 脚控制寄存器	I/O 脚控制寄存器
EFCR	额外特性寄存器	额外特性寄存器
DLL	除数锁存LSB(DLL) ^[3]	除数锁存LSB ^[3]
DLH	除数锁存MSB(DLH) ^[3]	除数锁存MSB ^[3]
EFR	增强型特性寄存器(EFR) ^[4]	增强型特性寄存器 ^[4]
XON1	Xon1 字 ^[4]	Xon1 字 ^[4]
XON2	Xon2 字 ^[4]	Xon2 字 ^[4]
XOFF1	Xoff1 字 ^[4]	Xoff1 字 ^[4]
XOFF2	Xoff2 字 ^[4]	Xoff2 字 ^[4]

[1] 只有当 EFR[4]置位时才可以修改 MCR[7]。

[2] 只有当 ERF[4]=1 且 MCR[2]=1（也就是说 EFR[4]和 MCR[2]为读/写使能时）才可以访问。

[3] 只有当 LCR[7]为逻辑 1 时才可以访问。

[4] 只有当 LCR 设为 10111111b (BFh) 时才可以访问。

表 10 SC16IS752/SC16IS762 内部寄存器

寄存器地址	寄存器	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0	R/W
通用寄存器集 ^[1]										
0x00	RHR	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0	R
0x00	THR	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0	W
0x01	IER	$\overline{\text{CTS}}$ 中断使能 ^[2]	$\overline{\text{RTS}}$ 中断使能 ^[2]	$\overline{\text{Xoff}}$ ^[2]	睡眠模式 ^[2]	Modem 状态中断	接收线状态中断	THR 空中断	Rx 数据可用中断	R/W
0x02	FCR	RX 触发点 (MSB)	RX 触发点 (LSB)	TX 触发点 (MSB) ^[2]	TX 触发点 (LSB) ^[2]	保留 ^[3]	TX FIFO 复位 ^[4]	RX FIFO 复位 ^[4]	FIFO 使能	W
0x02	IIR ^[5]	FIFO 使能	FIFO 使能	中断优先级位 4 ^[2]	中断优先级位 3 ^[2]	中断优先级位 2	中断优先级位 1	中断优先级位 0	中断状态	R
0x03	LCR	除数锁存使能	设置间隔	奇偶固定	偶选择	奇偶使能	停止位	字长度位 1	字长度位 0	R/W
0x04	MCR	时钟分频器 ^[2]	IrDA 模式使能 ^[2]	Xon Any ^[2]	环回使能	保留 ^[3]	TCR 和 TLR 使能 ^[2]	$\overline{\text{RTS}}$	$\overline{\text{DTR}}$ / (IO5)	R/W
0x05	LSR	FIFO 数据错误	THR 和 TSR 空	THR 空	间隔中断	帧错误	奇偶错误	溢出错误	接收器的数据	R
0x06	MSR	CD	RI	DSR	CTS	ΔCD	ΔRI	ΔDSR	ΔCTS	R
0x07	SPR	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0	R/W
0x06	TCR ^[6]	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0	R/W
0x07	TLR ^[6]	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0	R/W
0x08	TXLVL	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0	R
0x09	RXLVL	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0	R
0x0A	IODir ^[7]	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0	R/W
0x0B	IOState ^[7]	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0	R/W
0x0C	IOIntEna ^[7]	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0	R/W
0x0D	保留 ^[3]	保留 ^[3]	保留 ^[3]	保留 ^[3]	保留 ^[3]	保留 ^[3]	保留 ^[3]	保留 ^[3]	保留 ^[3]	
0x0E	IOControl ^[7]	保留 ^[3]	保留 ^[3]	保留 ^[3]	保留 ^[3]	UART 软件复位	I/O[3:0]或 $\overline{\text{RIB}}$, $\overline{\text{CDB}}$, $\overline{\text{DTRB}}$, $\overline{\text{DSRB}}$	I/O[7:4]或 $\overline{\text{RIA}}$, $\overline{\text{CDA}}$, $\overline{\text{DTRA}}$, $\overline{\text{DSRA}}$	锁存	R/W
0x0F	EFCR	IrDA 模式(慢/快) ^[8]	保留 ^[3]	自动 RS-485, RTS 输出反向	自动 RS-485, RTS 方向控制	保留 ^[3]	发送器禁能	接收器禁能	9 位模式使能	R/W

续上表

寄存器 地址	寄存器	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0	R/W
特殊寄存器集 ^[9]										
0x00	DLL	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0	R/W
0x01	DLH	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0	R/W
增强型寄存器集 ^[10]										
0x02	EFR	自动 $\overline{\text{CTS}}$	自动 $\overline{\text{RTS}}$	特殊字符 检测	使能增 强型功 能	软件流 控制位 3	软件流控制 位 2	软件流控制 位 1	软件流 控制位 0	R/W
0x04	Xon1	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0	R/W
0x05	Xon2	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0	R/W
0x06	Xoff1	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0	R/W
0x07	Xoff2	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0	R/W

[1] 这些寄存器仅当 LCR[7]=0 时可访问。

[2] 该位仅可在寄存器位 EFR[4]被使能时修改。

[3] 这些位为保留位且应设为 0。

[4] 接收 FIFO 或发送 FIFO 复位（通过 FCR[1:0]）后，用户必须在读或写数据到 RHR 和 THR 之前等待至少 XTAL1 的 2×Tperiod 时间。

[5] 在 IIR 寄存器上不应执行串行接口的突发读操作（那就是，在 I²C 总线上读多个元件而没有停止或重复的起始条件，或在 SPI 总线上读多个元件而没有拉低 $\overline{\text{CS}}$ 管脚）。

[6] 这些寄存器仅当 EFR[4]=1 且 MCR[2]=1 时可访问。

[7] 执行寄存器应用于两个通道中。

[8] SC16IS762 的 IrDA 模式为慢/快，SC16IS752 的模式为慢。

[9] 仅当 LCR[7]=1 且不为 0xBF 时特殊寄存器集可访问。

[10] 仅当 LCR=0xBF 时才可以访问增强型特性寄存器。

8.1 接收保存寄存器(RHR)

接收器部分由一个接收保存寄存器(RHR)和接收移位寄存器(RSR)组成。RHR 实际上是一个 64 字节 FIFO。RSR 接收 RX 端的串行数据。然后将数据转化为并行数据并转移到 RHR。线控制寄存器控制接收器部分。如果 FIFO 被禁能，则 FIFO 的单元 0 用来存储字符。

8.2 发送保存寄存器(THR)

发送器部分由一个发送保存寄存器(THR)和发送移位寄存器(TSR)组成。THR 实际上是一个 64 字节 FIFO。THR 接收数据并将其移入 TSR。然后在 TSR 中将其转化为串行数据并在 TX 端移出。如果 FIFO 被禁能，则 FIFO 仍用来存储字节。如果发生溢出则字符丢失。

8.3 FIFO 控制寄存器(FCR)

这是用作使能FIFO，清空FIFO，设置发送器和接收器触发点的只写寄存器。表 11所示为FIFO控制寄存器位设置。

表 11 FIFO 控制寄存器位描述

位	符号	描述
7:6	FCR[7] (MSB), FCR[6] (LSB)	RX 触发。这两位被用来设置 RX FIFO 的触发点。 00=8 字符 01=16 字符 10=56 字符 11=60 字符
5:4	FCR[5] (MSB), FCR[4] (LSB)	TX 触发。这两位被用来设置 TX FIFO 的触发点。 00=8 字符空间 01=16 字符空间 10=32 字符空间 11=56 字符空间 当 EFR[4]置位时，FCR[5:4]只可被修改和使能。这是因为发送触发点作为增强型功能使用。
3	FCR[3]	保留。
2	FCR[2] ^[1]	复位 TX FIFO。 逻辑 0=发送 FIFO 不复位（正常默认条件）。 逻辑 1=清空发送 FIFO 并复位 FIFO 电平逻辑（发送移位寄存器的内容不清除或改变）。FIFO 清空后该位将返回逻辑 0。
1	FCR[1] ^[1]	复位 RX FIFO。 逻辑 0=没有 FIFO 接收复位（正常默认条件）。 逻辑 1=清空接收 FIFO 和复位 FIFO 电平逻辑（接收移位寄存器的内容不清除或改变）。FIFO 清空后该位将返回逻辑 0。
0	FCR[0]	FIFO 使能。 逻辑 0=禁能发送和接收 FIFO（正常默认条件）。 逻辑 1=使能发送和接收 FIFO。

[1] FIFO 复位逻辑至少需要两个 XTAL1 时钟，因此，没有 XTAL1 时钟它们就不能被复位。

8.4 线控制寄存器(LCR)

线控制寄存器用来控制数据通信的格式。通过写寄存器的相应位来选择数据通信的字长度、停止位个数和奇偶性。表 12所示为线控制寄存器的位设置。

表 12 线控制寄存器位描述

位	符号	描述
7	LCR[7]	除数锁存使能。 逻辑 0=除数锁存禁能（正常默认条件）。 逻辑 1=除数锁存使能。
6	LCR[6]	间隔控制位。使能时，间隔控制位使得间隔条件被发送（TX 输出强制为逻辑 0 状态）。该条件将一直保持，直到通过清零 LCR[6]将其禁能。 逻辑 0=没有 TX 间隔条件（正常默认条件） 逻辑 1=强制发送器输出（TX）为逻辑 0，使出现线间隔条件时向远程接收器报警。
5	LCR[5]	奇偶固定。LCR[5]选择强制的奇偶格式（如果 LCR[3]=1）。 逻辑 0=不强制奇偶（正常默认条件） LCR[5]=逻辑 1 和 LCR[4]=逻辑 0: 发送和接收数据时奇偶位被强制为逻辑 1。 LCR[5]=逻辑 1 和 LCR[4]=逻辑 1: 发送和接收数据时奇偶位被强制为逻辑 0。
4	LCR[4]	奇偶类型选择。 逻辑 0=产生奇数格式（如果 LCR[3]=1）。 逻辑 1=产生偶数格式（如果 LCR[3]=1）。
3	LCR[3]	奇偶使能。 逻辑 0=无奇偶（正常默认条件）。 逻辑 1=在发送过程中产生一个奇偶位，接收器检测接收数据的奇偶性。
2	LCR[2]	停止位的个数。指定停止位的个数。 0~1 个停止位（字长度=5,6,7,8） 1~1.5 个停止位（字长度=5） 1~2 个停止位（字长度=6,7,8）
1:0	LCR[1:0]	字长度位 1, 0。这两位用来指定发送或接收的字长度（见表 15）。

表 13 LCR[5]奇偶选择

LCR[5]	LCR[4]	LCR[3]	奇偶选择
X	X	0	无奇偶
0	0	1	奇数
0	1	1	偶数
1	0	1	强制为 ‘1’
1	1	1	强制为 ‘0’

表 14 LCR[2]停止位长度

LCR[2]	字长度	停止位长度（位时间）
0	5, 6, 7, 8	1
1	5	$1\frac{1}{2}$
1	6, 7, 8	2

表 15 LCR[1:0]字长度

LCR[1]	LCR[0]	字长度（位）
0	0	5
0	1	6
1	0	7
1	1	8

8.5 线状态寄存器(LSR)

表 16所示为线状态寄存器的位设置。

表 16 线状态寄存器位描述

位	符号	描述
7	LSR[7]	FIFO 数据错误。 逻辑 0=无错误（正常默认条件）。 逻辑 1=接收 FIFO 数据中至少有一个奇偶错误、帧错误或间隔指示。当 FIFO 中不再出现错误时该位被清零。
6	LSR[6]	THR 和 TSR 为空。该位是发送空指示器。 逻辑 0=发送器保存和移位寄存器都不为空 逻辑 1=发送器保存和移位寄存器都为空
5	LSR[5]	THR 为空。该位是发送保存寄存器空指示器。 逻辑 0=发送保存寄存器不为空 逻辑 1=发送保存寄存器为空。如果 TX FIFO 使能，那么主机可将多达 64 字符的数据装入 THR。
4	LSR[4]	间隔中断。 逻辑 0=无间隔条件（正常默认条件）。 逻辑 1=出现间隔条件且相关的字符为 00h (RX 在一个字符时间帧内持续为低电平)。
3	LSR[3]	帧错误。 逻辑 0=正在从 RX FIFO 中读取的数据无帧错误（正常默认条件）。 逻辑 1=正在从 RX FIFO 中读取的数据出现帧错误（接收的数据中没有有效的停止位）。
2	LSR[2]	奇偶错误。 逻辑 0=无奇偶错误（正常默认条件）。 逻辑 1=正在从 RX FIFO 中读取的数据出现奇偶错误。
1	LSR[1]	超时错误。 逻辑 0=无超时错误（正常默认条件）。 逻辑 1=出现超时错误。
0	LSR[0]	接收器中的数据。 逻辑 0=接收 FIFO 中无数据（正常默认条件）。 逻辑 1=RX FIFO 中至少有一个字符。

读 LSR 时，LSR[4:2]反映 RX FIFO 顶端字符（要读取的下一个字符）的错误位(BI, FE, PE)。因此，通过先读取 LSR 然后再读取 RHR 来识别字符中的错误。

当在 RX FIFO 中任何地方出现错误时 LSR[7]置位，且仅当 FIFO 中不再有剩余的错误时 LSR[7]清零。

8.6 Modem 控制寄存器(MCR)

MCR控制着模式、数据集或仿真modem的外围器件间的连接。表 17给出modem控制寄存器的位设置。

表 17 modem 控制寄存器位描述

位	符号	描述
7	MCR[7] ^[1]	时钟分频器。 逻辑 0=1 分频-时钟输入 逻辑 1=4 分频-时钟输入
6	MCR[6] ^[1]	IrDA 模式使能。 逻辑 0=正常 UART 模式。 逻辑 1=IrDA 模式。
5	MCR[5] ^[1]	Xon Any。 逻辑 0=禁能 Xon Any 功能。 逻辑 1=使能 Xon Any 功能。
4	MCR[4]	使能环回模式。 逻辑 0=正常操作模式。 逻辑 1=使能局部环回模式（内部）。在该模式下，MCR[1:0]信号回送到 MSR[4:5]且 TX 输出内部回送到 RX 输入。
3	MCR[3]	保留。
2	MCR[2]	TCR 和 TLR 使能。 逻辑 0=禁能 TCR 和 TLR 寄存器 逻辑 1=使能 TCR 和 TLR 寄存器
1	MCR[1]	$\overline{\text{RTS}}$ 逻辑 0=强制 $\overline{\text{RTS}}$ 输出无效（高电平）。 逻辑 1=强制 $\overline{\text{RTS}}$ 输出有效（低电平）。在环回模式下，控制 MSR[4]。如果自动 $\overline{\text{RTS}}$ 使能，那么由硬件流控制来控制 $\overline{\text{RTS}}$ 输出。
0	MCR[0]	$\overline{\text{DTR}}$ 。如果 GPIO5 或 GPIO1 通过 IOControl 寄存器位 1 或位 2 被选择用作 $\overline{\text{DTR}}$ modem 管脚，那么 $\overline{\text{DTR}}$ 管脚的状态可如下控制。写 IOState 位 5 或位 1 将不会对 $\overline{\text{DTR}}$ 管脚有任何影响。 逻辑 0=强制 $\overline{\text{DTR}}$ 输出无效（高电平）。 逻辑 1=强制 $\overline{\text{DTR}}$ 输出有效（低电平）。

[1] MCR[7:5]和 MCR[2]只可在 EFR[4]置位时被修改，那就是，EFR[4]为写使能。

8.7 Modem 状态寄存器(MSR)

该 8 位寄存器给出了从modem，数据集或外围器件到主机的控制线的当前状态。它也指示modem控制输入状态改变时的情况。表 18给出了每个通道modem状态寄存器的位设置。

表 18 modem 状态寄存器位描述

位	符号	描述
7	MSR[7]	CD (高电平有效, 逻辑 1)。如果 GPIO6 或 GPIO2 通过 IOControl 寄存器位 1 或位 2 选择作为 $\overline{\text{CD}}$ modem 管脚, 则该位可读出 $\overline{\text{CD}}$ 管脚的状态。该位是 $\overline{\text{CD}}$ 输入的补码。读 IOState 位 6 或位 2 不反映 $\overline{\text{CD}}$ 脚的真实状态。
6	MSR[6]	RI (高电平有效, 逻辑 1)。如果 GPIO7 或 GPIO3 通过 IOControl 寄存器位 1 或位 2 选择作为 $\overline{\text{RI}}$ modem 管脚, 则该位可读出 $\overline{\text{RI}}$ 管脚的状态。该位是 $\overline{\text{RI}}$ 输入的补码。读 IOState 位 7 或位 3 不反映 $\overline{\text{RI}}$ 脚的真实状态。
5	MSR[5]	DSR (高电平有效, 逻辑 1)。如果 GPIO4 或 GPIO0 通过 IOControl 寄存器位 1 或位 2 选择作为 $\overline{\text{DSR}}$ modem 管脚, 则该位可读出 $\overline{\text{DSR}}$ 管脚的状态。该位是 $\overline{\text{DSR}}$ 输入的补码。读 IOState 位 4 或位 0 不反映 $\overline{\text{DSR}}$ 脚的真实状态。
4	MSR[4]	CTS (高电平有效, 逻辑 1)。该位是 $\overline{\text{CTS}}$ 输入的补码。
3	MSR[3]	ΔCD 。表示 $\overline{\text{CD}}$ 输入状态已改变。该位在读取时被清零。
2	MSR[2]	ΔRI 。表示 $\overline{\text{RI}}$ 输入的状态从低电平变为高电平。该位在读取时被清零。
1	MSR[1]	ΔDSR 。表示 $\overline{\text{DSR}}$ 输入状态已改变。该位在读取时被清零。
0	MSR[0]	ΔCTS 。表示 $\overline{\text{CTS}}$ 输入状态已改变。该位在读取时被清零。

注: 主要输入 $\overline{\text{RI}}$, $\overline{\text{CD}}$, $\overline{\text{CTS}}$, $\overline{\text{DSR}}$ 全为低电平有效。

8.8 中断使能寄存器(IER)

中断使能寄存器 (IER) 使能 6 种中断类型, 这些中断类型包括接收器错误、RHR 中断、THR 中断、modem 状态、接收的 Xoff 或 $\overline{\text{CTS}}$ / $\overline{\text{RTS}}$ 从低到高的状态变化。激活 $\overline{\text{IRQ}}$ 输出信号来响应中断产生。表 19 给出中断使能寄存器的位设置。

表 19 中断使能寄存器位描述

位	符号	描述
7	IER[7] ^[1]	$\overline{\text{CTS}}$ 中断使能。 逻辑 0 = 禁能 $\overline{\text{CTS}}$ 中断 (正常默认条件)。 逻辑 1 = 使能 $\overline{\text{CTS}}$ 中断。
6	IER[6] ^[1]	$\overline{\text{RTS}}$ 中断使能。 逻辑 0 = 禁能 $\overline{\text{RTS}}$ 中断 (正常默认条件)。 逻辑 1 = 使能 $\overline{\text{RTS}}$ 中断。
5	IER[5] ^[1]	Xoff 中断。 逻辑 0 = 禁能 Xoff 中断 (正常默认条件)。 逻辑 1 = 使能 Xoff 中断。
4	IER[4] ^[1]	睡眠模式。 逻辑 0 = 禁能睡眠模式 (正常默认条件)。 逻辑 1 = 使能睡眠模式。详见 7.6 节“睡眠模式”。

续上表

位	符号	描述
3	IER[3]	Modem 状态中断。 逻辑 0=禁能 modem 状态寄存器中断（正常默认条件）。 逻辑 1=使能 modem 状态寄存器中断。 注： 关于如何设定管脚为 modem 管脚，详见 IOControl 寄存器位 1 或位 2（在表 30）。
2	IER[2]	接收线状态中断。 逻辑 0=禁能接收器线状态中断（正常默认条件）。 逻辑 1=使能接收器线状态中断。
1	IER[1]	发送保存寄存器中断。 逻辑 0=禁能 THR 中断（正常默认条件）。 逻辑 1=使能 THR 中断。
0	IER[0]	接收保存寄存器中断。 逻辑 0=禁能 RHR 中断（正常默认条件）。 逻辑 1=使能 RHR 中断。

[1] IER[7:4]只可在 EFR[4]置位时被修改，那就是，EFR[4]为写使能。如果 THR 低于阈值，那么重新使能 IER[1]将不会产生新的中断

8.9 中断识别寄存器(IIR)

IIR是以优先级方式提供中断源的只读 8 位寄存器。表 20给出中断识别寄存器的位设置。

表 20 中断识别寄存器位描述

位	符号	描述
7:6	IIR[7:6]	反映 FCR[0]的内容。
5:1	IIR[5:1]	5 位译码中断。见表 21。
0	IIR[0]	中断状态。 逻辑 0=中断等待处理。 逻辑 1=没有中断等待处理。

表 21 中断源

优先级	IIR[5]	IIR[4]	IIR[3]	IIR[2]	IIR[1]	IIR[0]	中断源
1	0	0	0	1	1	0	接收器线状态错误
2	0	0	1	1	0	0	接收器超时中断
2	0	0	0	1	0	0	RHR 中断
3	0	0	0	0	1	0	THR 中断
4	0	0	0	0	0	0	Modem中断 ^[1]
5	1	1	0	0	0	0	输入脚的状态变化 ^[1]
6	0	1	0	0	0	0	接收的 Xoff 信号/特殊字符
7	1	0	0	0	0	0	$\overline{\text{CTS}}$ 、 $\overline{\text{RTS}}$ 状态变化从有效（低电平）到无效（高电平）

[1] modem 中断状态必须通过 MSR 寄存器读出且 GPIO 中断状态必须通过 IOState 寄存器读出。

8.10 增强型特性寄存器(EFR)

该 8 位寄存器使能或禁能UART的增强型特性。表 22给出增强型特性寄存器的位设置。

表 22 增强型特性寄存器位描述

位	符号	描述
7	EFR[7]	$\overline{\text{CTS}}$ 流控制使能。 逻辑 0 = $\overline{\text{CTS}}$ 流控制禁能（正常默认条件）。 逻辑 1 = $\overline{\text{CTS}}$ 流控制使能。当在 $\overline{\text{CTS}}$ 管脚上检测到高电平信号时发送将停止。
6	EFR[6]	$\overline{\text{RTS}}$ 流控制使能。 逻辑 0 = $\overline{\text{RTS}}$ 流控制禁能（正常默认条件）。 逻辑 1 = $\overline{\text{RTS}}$ 流控制使能。当到达接收器 FIFO HALT 触发点 TCR[3:0]时 $\overline{\text{RTS}}$ 脚变高，且当到达接收器 FIFO RESUME 发送触发点 TCR[7:4]时 $\overline{\text{RTS}}$ 脚变低。
5	EFR[5]	特殊字符检测。 逻辑 0 = 特殊字符检测禁能（正常默认条件）。 逻辑 1 = 特殊字符检测使能。将接收的数据与 Xoff2 数据比较。如果匹配发生，接收的数据传输到 FIFO 且 IIR[4]被设为逻辑 1 来表示已检测到一个特殊字符。
4	EFR[4]	增强型功能使能位。 逻辑 0 = 禁能增强型功能并写 IER[7:4], FCR[5:4], MCR[7:5]。 逻辑 1 = 使能增强型功能 IER[7:4], FCR[5:4]和 MCR[7:5]使其可被修改。
3:0	EFR[3:0]	可通过设定这些位来选择软件流控制的组合。见表 3“软件流控制选项(EFR[3:0])”。

8.11 除数寄存器(DLL, DLH)

除数寄存器为 2 个 8 位寄存器，它们存储波特率发生器中波特率时钟产生的 16 位除数。DLH 存储除数的最高部分。DLL 存储除数的最低部分。

需要注意的是 DLL 和 DLH 只能在睡眠模式使能之前（在 IER[4]置位之前）被写入。

8.12 发送控制寄存器(TCR)

该 8 位寄存器用于在硬件/软件流控制过程中存储RX FIFO阈值来停止/启动发送。表 23给出发送控制寄存器的位设置。

表 23 发送控制寄存器位描述

位	符号	描述
7:4	TCR[7:4]	RX FIFO 的恢复触发点
3:0	TCR[3:0]	RX FIFO 的中止传输触发点

TCR 触发点可从 0 到 60 字符中得到（以 4 为间隔）。

备注: 当 EFR[4]=1 且 MCR[2]=1 时才能写 TCR。程序员必须编程 TCR 以使 TCR[3:0] > TCR[7:4]。没有内置的硬件检测来确保是否符合这种条件。同时，必须在自动 $\overline{\text{RTS}}$ 或软件流控制使能之前用这种条件编程 TCR，来避免器件的错误操作。

8.13 触发点寄存器(TLR)

该 8 位寄存器存储用来产生中断的发送和接收FIFO触发点。设定从 4 到 60 的触发点(以 4 为间隔)。表 24 给出触发点寄存器的位设置。

表 24 触发点寄存器位描述

位	符号	描述
7:4	TLR[7:4]	RX FIFO 触发点 (4 到 60), 可用的字符数
3:0	TLR[3:0]	TX FIFO 触发点 (4 到 60), 可用的空间数

备注: 当 EFR[4]=1 且 MCR[2]=1 时才能写 TLR。如果 TLR[3:0]或 TLR[7:4]为逻辑 0, 那么通过 FIFO 控制寄存器(FCR)的可选触发点用于发送和接收 FIFO 触发点。可使用从 4 字符到 60 字符的触发点 (以 4 为间隔)。TLR 应被编程为 N/4, 其中 N 为所需的触发点。

当 TLR 中的触发点设置为 0 时, SC16IS752/SC16IS762 使用 FCR 中定义的触发点设置。如果 TLR 含有非零的触发点值, 那么不使用 FCR 中定义的触发点。这种情况应用于发送 FIFO 和接收 FIFO 触发点设置。

当 TLR 用于 RX 触发点控制时, FCR[7:6]应保持为默认状态 ‘00’。

8.14 发送器 FIFO 电平寄存器(TXLVL)

该寄存器为只读寄存器, 它报导发送 FIFO 中可用的空间数。

表 25 发送器 FIFO 电平寄存器位描述

位	符号	描述
7	-	不使用; 设为 0
6:0	TXLVL[6:0]	TX FIFO 中可用的空间数, 从 0 (0x00)到 64 (0x40)。

8.15 接收器 FIFO 电平寄存器(RXLVL)

该寄存器为只读寄存器, 它报导接收 FIFO 的填充电平, 那就是, RXFIFO 中的字符数。

表 26 接收器 FIFO 电平寄存器位描述

位	符号	描述
7	-	不使用; 设为 0
6:0	RXLVL[6:0]	RX FIFO 中存储的字符数, 从 0 (0x00)到 64 (0x40)。

8.16 可编程的 I/O 脚方向寄存器 (IODir)

该寄存器用于编程 I/O 脚方向。位 0~位 7 控制着 GPIO0~GPIO7。

表 27 IODir 寄存器位描述

位	符号	描述
7:0	IODir	设置 GPIO 脚[7:0]为输入或输出。 0=输入 1=输出

8.17 可编程的 I/O 脚状态寄存器(IOState)

当‘读取’时，该寄存器返回所有 I/O 脚的实际状态。当‘写入’时，每个寄存器位将被传输到设定为输出的相应 I/O 脚。

表 28 IOState 寄存器位描述

位	符号	描述
7:0	IOState	写该寄存器： 在输出管脚上设置逻辑电平 0=设置输出管脚为 0 1=设置输出管脚为 1 读该寄存器： 返回所有管脚的状态

8.18 I/O 中断使能寄存器(IOIntEna)

当 I/O 配置为输入发生改变时，该寄存器使能中断。如果 GPIO[7:4]或 GPIO[3:0]被设定为 modem 管脚，那么必须通过 IER[3]使能它们产生中断。在这种情况下，IOIntEna 将不影响 GPIO[7:4]或 GPIO[3:0]。

表 29 IOIntEna 寄存器位描述

位	符号	描述
7:0	IOIntEna	输入中断使能。 0=输入管脚的变化将不会产生中断 1=输入管脚的变化将产生中断

8.19 I/O 控制寄存器 (IOControl)

表 30 IOControl 寄存器位描述 n

位	符号	描述
7:4	保留	这些位保留为将来使用。
3	SReset	软件复位。写该位将复位器件。一旦器件复位，则该位自动设为‘0’。
2	GPIO[3:0] 或 \overline{RIB} , \overline{CDB} , \overline{DTRB} , \overline{DSRB}	该位设定 GPIO[3:0]为 I/O 脚或作为 modem 的管脚。 0=I/O 脚 1=GPIO[3:0]模拟 \overline{RIB} , \overline{CDB} , \overline{DTRB} , \overline{DSRB}

续上表

位	符号	描述
1	GPIO[7:4] 或 \overline{RIA} , \overline{CDA} , \overline{DTRA} , \overline{DSRA}	该位设定 GPIO[7:4]为 I/O 脚或作为 modem 的管脚。 0=I/O 脚 1=GPIO[7:4]模拟 \overline{RIA} , \overline{CDA} , \overline{DTRA} , \overline{DSRA}
0	IOLatch	使能/禁能输入锁存。 0=输入值没有被锁存。任何输入的改变将产生中断。读输入寄存器清除中断。如果在读输入寄存器之前输入返回到其初始的逻辑状态，那么中断被清除。 1=输入值被锁存。输入的改变将产生中断且输入逻辑值被装入输入状态寄存器(IOSState)的相应位中。读 IOSState 寄存器清除中断。如果在读中断寄存器之前输入管脚返回到其初始的逻辑状态，那么中断不被清除且 IOSState 寄存器的相应位保持初始化中断的逻辑值。

备注: 作为 I/O 脚, 方向、状态和 GPIO 的中断使能由下列的寄存器控制: IODir, IOSState, IOIntEna 和 IOControl。 \overline{CD} , \overline{RI} , \overline{DSR} 脚的状态将不会在 MSR[7:5]或 MSR[3:1]中反映出来, 且在这三个管脚上的任何状态变化将不会触发 modem 状态中断 (尽管通过 IER[3]使能), 且 \overline{DTR} 管脚的状态不能由 MCR[0]来控制。

作为 modem 的 \overline{CD} , \overline{RI} , \overline{DSR} 脚, 这三个管脚的输入状态可从 MSR[7:5]和 MSR[3:1]中读出, 且 \overline{DTR} 管脚的状态可以由 MCR[0]来控制。同时, 如果 modem 状态中断位 (IER[3]) 被使能, \overline{CD} , \overline{RI} , \overline{DSR} 脚上的状态变化将触发 modem 中断。IODir, IOSState 和 IOIntEna 寄存器将不会对这三个管脚有任何影响。

8.20 额外特性控制寄存器(EFCR)

表 31 额外特性控制寄存器位描述

位	符号	描述
7	IrDA 模式	IrDA 模式 0=IrDA 版本 1.0, 3/16 脉冲比率, 数据率高达 115.2kbit/s 1=IrDA版本 1.1, 1/4 脉冲比率, 数据率高达 1.152Mbit/s ^[1]
6	-	保留
5	RTSInver	在 RS-485 模式中翻转 \overline{RTS} 信号。 0: 发送过程中 $\overline{RTS}=0$ 且接收过程中 $\overline{RTS}=1$ 1: 发送过程中 $\overline{RTS}=1$ 且接收过程中 $\overline{RTS}=0$
4	RTSCon	使能发送器来控制 \overline{RTS} 管脚。 0: 发送器不控制 \overline{RTS} 管脚 1: 发送器控制 \overline{RTS} 管脚
3	-	保留
2	TXDisable	禁能发送器。UART 在发送管脚上不发送串行数据, 但发送 FIFO 将继续接收主机的数据直到接收满。在发送器变为禁能状态之前, TSR 中的任何数据将被发送出去。 0: 发送器被使能 1: 发送器被禁能

续上表

位	符号	描述
1	RXDisable	禁能接收器。一旦该位被设为 1，UART 将立即停止接收数据，且 TSR 中的任何数据将被发送到接收 FIFO。建议用户在接收过程中不要设置该位。 0: 接收器被使能 1: 接收器被禁能
0	9 位模式	使能 9 位或多点模式 (RS-485) 0: 正常 RS-232 模式 1: 使能 RS-485 模式

[1] 仅用于 SC16IS762。

9.RS-485 特性

9.1 自动 RS-485 $\overline{\text{RTS}}$ 控制

通常 $\overline{\text{RTS}}$ 管脚由 MCR 位 1 控制，或如果硬件流控制被使能， $\overline{\text{RTS}}$ 管脚的逻辑状态由硬件流控制电路控制。EFCR 寄存器位 4 将优先考虑其它两种模式；一旦该位置位，发送器将控制 $\overline{\text{RTS}}$ 管脚的状态。一旦主机写数据到发送 FIFO，发送器将自动使 $\overline{\text{RTS}}$ 管脚（逻辑 0）有效；一旦发送了数据的最后位，发送器将使 $\overline{\text{RTS}}$ 管脚（逻辑 1）无效。

要使用自动 RS-485 $\overline{\text{RTS}}$ 模式，软件将不得不禁能硬件流控制功能。

9.2 RS-485 $\overline{\text{RTS}}$ 输出翻转

如果 UART 置于自动 RS-485 $\overline{\text{RTS}}$ 模式下，那么 EFCR 位 5 翻转 $\overline{\text{RTS}}$ 管脚的极性。当发送器有数据要发送时它使 $\overline{\text{RTS}}$ 管脚（逻辑 1）无效，且当数据的最后位已发送时发送器使 $\overline{\text{RTS}}$ 管脚（逻辑 0）有效。

9.3 自动 RS-485

EFCR 位 0 用于使能 RS-485 模式（多点或 9 位模式）。在这种操作模式下，‘主机’方先发送地址字符，然后再发送用来寻址‘从机’方的数据字符。如果接收字符为地址字符（奇偶位=1），那么从机方检测接收的数据并中断控制器。

要使用自动 RS-485 $\overline{\text{RTS}}$ 模式，软件将不得不禁能硬件流控制功能。

9.3.1 正常多点模式

使能 EFCR 中的 9 位模式（位 0），而不是特殊字符检测（EFCR 位 5）。接收器设为强制奇偶 0 (LCR[5:3]=111)来检测地址字节。

开始禁能接收器后，接收器忽略所有数据字节（奇偶位=0）直至接收到地址字节（奇偶位=1）。该地址字节将使 UART 设置奇偶错误。UART 将产生线状态中断（这时 IER 位 2 将被设为‘1’），且同时在 RX FIFO 中放置该地址字节。在控制器检测字节后，它必须决定是否使能接收器；如果地址字节寻址其 ID 地址则它应使能接收器，且如果地址字节不寻址其 ID 地址则它必须不可使能接收器。

如果控制器使能接收器，那么在控制器接收到‘主机’方完整的信息后，接收器将接收后续的数据直至被控制器禁能。如果在接收到‘主机’方的信息后控制器不禁能接收器，那么接收器将在接收另一个地址字节时产生奇偶错误。然后控制器决定地址字节是否寻址其ID地址，如果不是，控制器可以禁能接收器。如果地址字节寻址‘从机’ID地址，那么控制器不执行进一步的操作；接收器将接收后续的数据。

9.3.2 自动地址检测

如果使能特殊字符检测（置位 EFR[5]且 XOFF2 包含地址字节），那么接收器将尝试检测匹配 XOFF2 中编程字符的地址字节。如果接收字节为数据字节或地址字节不匹配 XOFF2 中的编程字符，那么接收器将放弃这些数据。在接收匹配 XOFF2 字符的地址字节时，如果还没有使能则接收器将自动使能，且地址字符与奇偶位（而不是奇偶错误位）一起放入 RX FIFO。同时接收器产生一个线状态中断（这时 IER 位 2 必须设为 1）。接收到‘主机’方的信息后，接收器将从‘主机’方接收后续的数据直至被控制器禁能。

如果接收到另一个地址字节且该地址字节不匹配 XOFF2 字符，接收器将被自动禁能且忽略地址字节。如果地址字节匹配 XOFF2 字符，接收器将在 RX FIFO 中放置该字节和奇偶错误位中的奇偶位(LSR[2])。

10. I²C 总线操作

I²C 总线的两条线为串行数据线(SDA)和串行时钟线(SCL)。两条线都通过一个上拉电阻连接到正极电源，且在总线不忙时保持高电平。无论器件是微计算机、LCD驱动器、存储器还是键盘接口，根据器件的功能，每个器件都被唯一的地址识别并可用作发送器或接收器。产生信息或数据的器件为发送器，且接收信息或数据的器件为接收器。很明显，像LCD驱动器那样的无源功能仅可作为接收器，而微控制器或存储器都可发送和接收数据。

10.1 数据传输

在每个时钟脉冲过程中传输一个数据位（见图 12）。时钟脉冲的高电平期间SDA线上的数据必须保持稳定来使其有效。这时数据线上的变化将被看成为控制信号。时钟信号(SCL)为高时数据线(SDA)上高到低的跳变表示起始条件，而SCL为高时SDA上低到高的跳变表示停止条件（见图 13）。在起始条件后总线被认为忙，并且在停止条件后的特定时间间隔处再次空闲。总是由主机产生起始和停止条件。

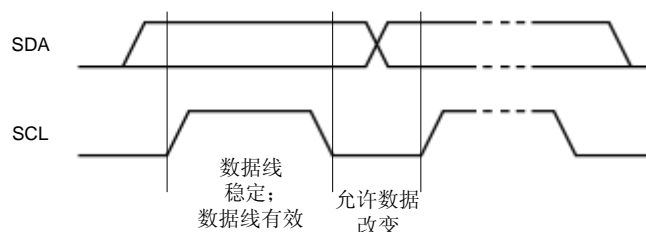


图 12 I²C总线上的位传输

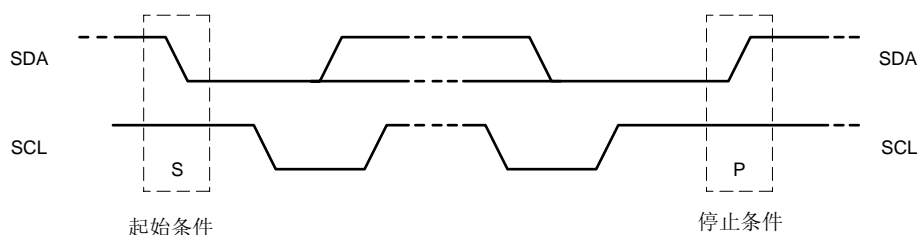


图 13 起始和停止条件

从发送器到接收器的起始和停止条件之间传输的数据字节数不限。每个字节(必须为 8 位长)首先与最高位、后面接着是应答位一起串行传输(见图 14)。与应答位有关的时钟脉冲由主机产生。在应答时钟脉冲期间应答的器件必须拉低SDA线,而发送的器件将释放该脉冲(见图 15)。

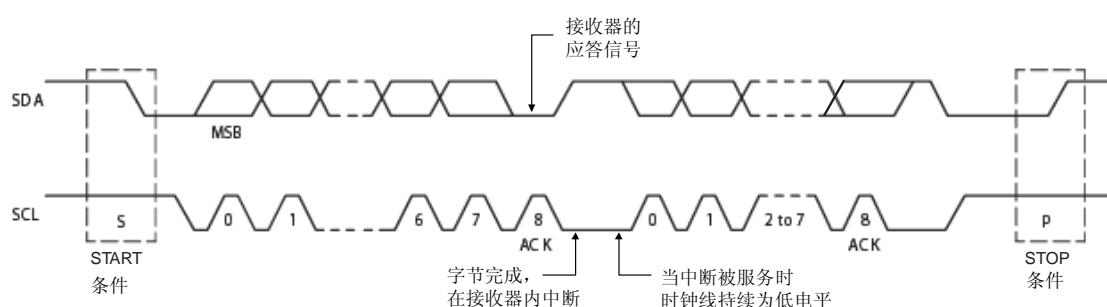


图 14 I²C总线上的数据传输

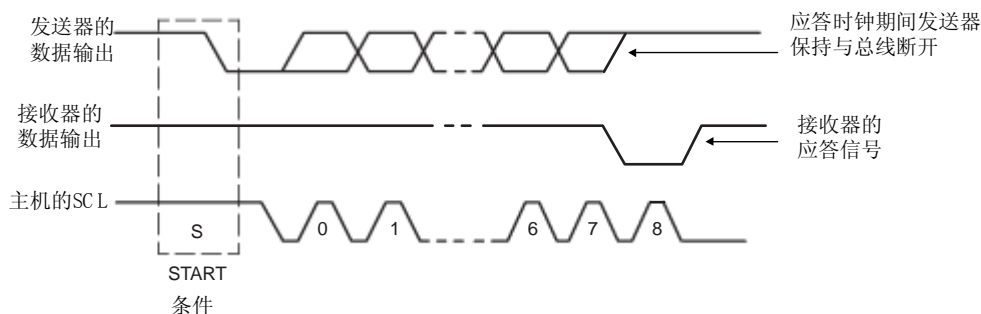


图 15 I²C总线上的应答

从机接收器必须在接收到每个字节后产生一个应答,并且主机必须在接收到从机发送器之外计时的每个字节后产生一个应答。当设计系统时,需要考虑没有接收到应答的情况。例如,当寻址的器件在实时操作中忙时出现这种情况。此时,主机在适当的‘超时’后应通过产生停止条件来中止传输,而允许产生其它的传输。在多主机系统中,这些‘其它传输’可以由其它主机或这个相同的主机来启动。

‘每字节后应答’的规则有两种例外的情况。当主机为接收器时出现第一种例外情况:它必须指示到发送器的数据的结束,而不表示已在从机之外计时的最后一个字节的应答。主机产生的与时钟相关的应答仍然发生,但 SDA 线将不会被拉低。为了表示这种有效和有意缺少的应答,我们将这种特殊条件称为‘否定应答’。

第二种例外情况是当从机不能再接收其它的数据字节时它将发送一个否定应答。这种情况在不能接收的传输后发生。

10.2 寻址和传输格式

总线上的每个器件都有其自身唯一的地址。在总线上发送任何数据之前，主机在总线上发送这次传输要访问的从机地址。如果一个带匹配地址的正常操作的从机在网络上出现，它当然将应答主机的寻址。在起始条件后，由主机发送的第一个字节来完成寻址。

在网络上的地址为 7 位长，作为地址字节的最高位出现。最后一位是方向位 (R/\overline{W})。0 表示主机正在发送（‘写’）而 1 表示主机请求数据（‘读’）。完整的数据传输包括一个表示‘写’的地址字节和两个数据字节，如图 16 所示。

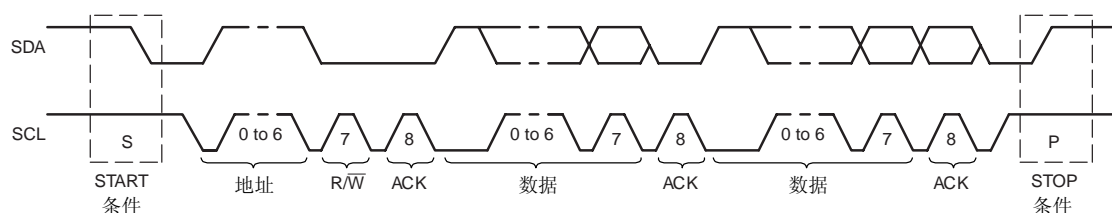


图 16 完整的数据传输

当发送地址时，系统中的每个器件在带有其自身地址的 START 后比较最初的 7 个位。如果发生匹配，器件将认为其自身由主机寻址，且将发送一个应答。根据 R/\overline{W} 位，器件也可确定它是否在该传输中被指定为从机接收器还是从机发送器。

I²C 总线网络的每个节点具有唯一的 7 位地址。微控制器的地址完全可编程，而外设器件通常具有固定且可编程的地址部分。

当主机仅与一个器件通信时，数据传输按照图 16 的格式进行，其中 R/\overline{W} 位可指示任意的方向。在完成传输和发布停止条件后，如果主机想在网络上寻址一些其它的器件，它可通过发布一个新的 START 来启动另一个传输。

主机还可以通过使用‘Repeated START’与几种不同的器件通信。操作的最后一个字符传输完后，包括其应答（或否定应答），主机将发布另一个 START，后面跟着地址字节和数据（而不影响 STOP）。主机可与许多不同的器件通信，组成‘读’和‘写’。在最后的传输发生后，主机发布 STOP 并释放总线。可执行的数据格式如图 17 所示。注意重复的 START 允许从机和方向的同时改变，而无需释放总线。我们将在后面看到即使在处理一个器件时，也可以容易获取方向改变的特性。

在单个主机系统中，重复的 START 机制可能比用 STOP 终止每个传输然后又再次启动更有效。在多主机环境中，决定哪个格式更有效可能会变得更复杂，因为当主机使用重复的 START 时占用总线一段长的时间，从而阻止了其它器件启动传输。

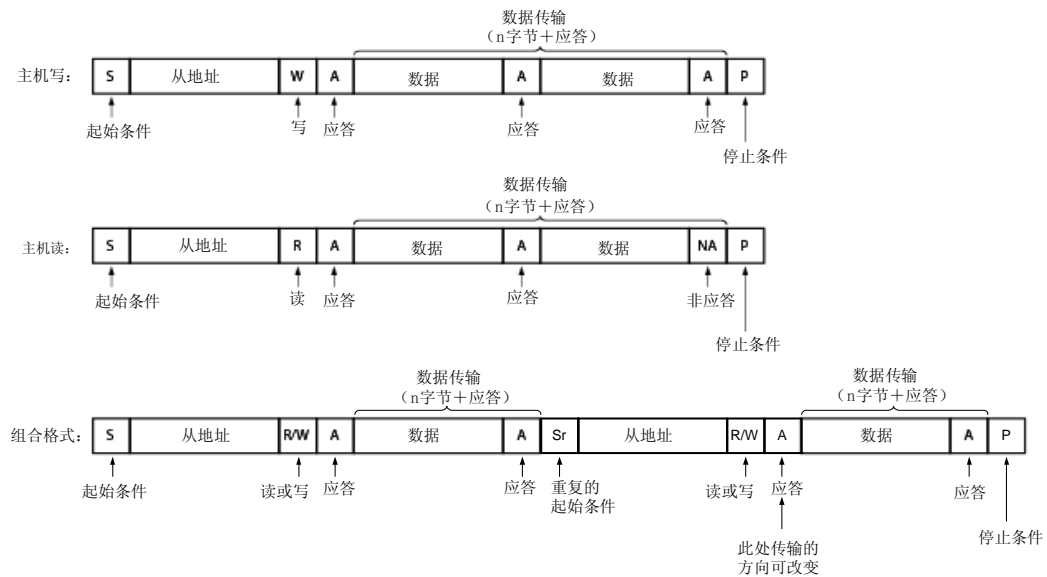


图 17 I²C总线格式

10.3 寻址

在发送或接收任何数据之前，主机必须通过SDA线发送接收器的地址。起始条件后的第一个字节包含从器件的地址和读/写位。表 32给出如何使用A1 和A0 管脚选择SC16IS752/SC16IS762 的地址。例如，如果这两个管脚连接到V_{DD}，那么SC16IS752/SC16IS762 的地址设为 0x90，且主机通过这个地址与其通信。

表 32 SC16IS752/SC16IS762 地址映射

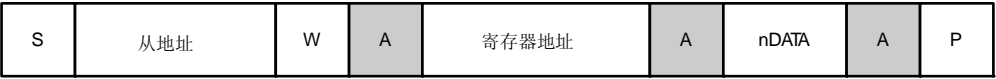
A1	A0	SC16IS752/ SC16IS762 I ² C地址 (hex) ^[1]
V _{DD}	V _{DD}	0x90 (1001 000X)
V _{DD}	V _{SS}	0x92 (1001 001X)
V _{DD}	SCL	0x94 (1001 010X)
V _{DD}	SDA	0x96 (1001 011X)
V _{SS}	V _{DD}	0x98 (1001 100X)
V _{SS}	V _{SS}	0x9A (1001 101X)
V _{SS}	SCL	0x9C (1001 110X)
V _{SS}	SDA	0x9E (1001 111X)
SCL	V _{DD}	0xA0 (1010 000X)
SCL	V _{SS}	0xA2 (1010 001X)
SCL	SCL	0xA4 (1010 010X)
SCL	SDA	0xA6 (1010 011X)
SDA	V _{DD}	0xA8 (1010 100X)
SDA	V _{SS}	0xAA (1010 101X)
SDA	SCL	0xAC (1010 110X)
SDA	SDA	0xAE (1010 111X)

[1] X=逻辑 ‘0’ 用于写周期；X=逻辑 ‘1’ 用于读周期。

10.4 子地址的使用

当主机与 SC16IS752/SC16IS762 通信时,它必须在跟随从地址字节后的字节中发送一个子地址。对于单个字节传输,该子地址是主机想要访问的字内部地址;或对于多个字节传输,子地址是单元序列的开始。子地址是一个 8 位字节。不同于器件地址,子地址不含有方向(R/ \overline{W})位,并且像总线上传输的任何字节一样后面必须跟着一个应答。

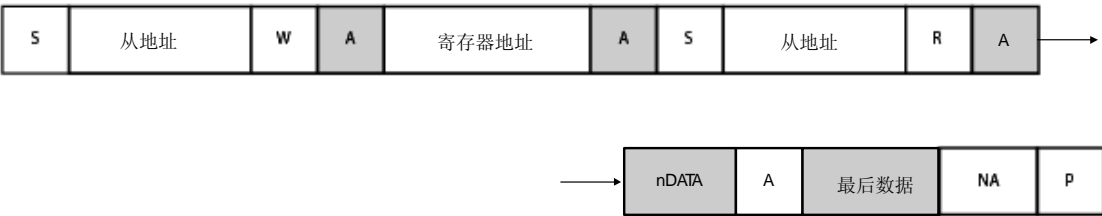
寄存器写周期如图 18 所示。START 后面跟着一个带有方向位设为‘写’的从地址字节,一个子地址字节,许多数据字节和一个STOP(停止)信号。子地址表示主机想访问的寄存器,及其后面的数据字节将被一个接一个地写入子地址单元。



白色模块: 主机到 SC16IS752/SC16IS762
灰色模块: SC16IS752/SC16IS762 到主机

图 18 主机写到从机

寄存器读周期(见图 19)以类似的方式开始,主机发送方向位设为WRITE(‘写’)的从地址(后面为子地址)。接着,为了翻转传输的方向,主机发布一个重复START,后面再次跟着器件地址,但这时方向位设为READ(‘读’)。内部子地址处开始的数据字节将在器件之外被计时,每个字节后面都跟着主机产生的应答。读周期的最后一个字节后面跟着否定应答,表示传输的结束。周期由STOP信号终止。



白色模块: 主机到 SC16IS752/SC16IS762
灰色模块: SC16IS752/SC16IS762 到主机

图 19 主机从从机读

表 33 寄存器地址字节 (I²C)

位	名称	功能
7	-	不使用
6:3	A[3:0]	UART 的内部寄存器选择
2:1	CH1, CH0	通道选择。 00=通道 A 01=通道 B 10=保留 11=保留
0	-	不使用

11.SPI 操作

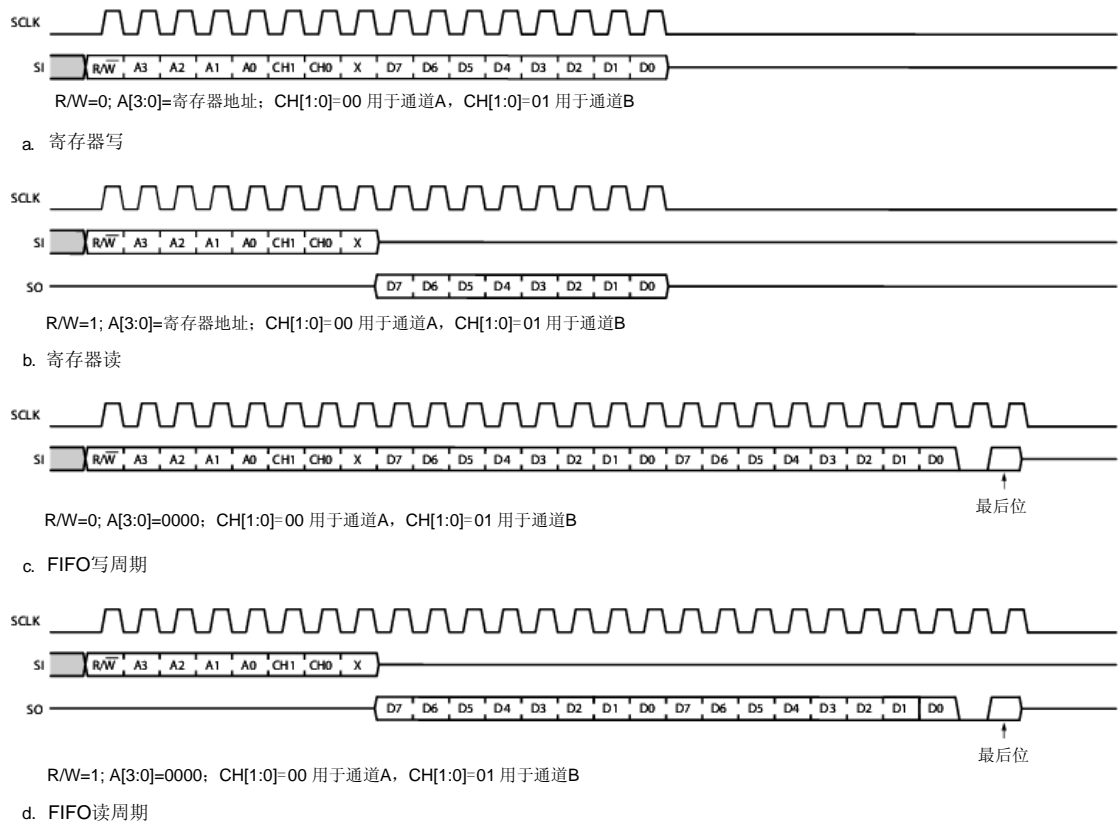


图 20 SPI 操作

表 34 寄存器地址字节 (SPI)

位	名称	功能
7	R/ \overline{W}	读/写。 1=从 UART 读 0=写入 UART
6:3	A[3:0]	UART 的内部寄存器选择
2:1	CH1, CH0	通道选择。 00=通道 A 01=通道 B 10=保留 11=保留
0	-	不使用

12. 极限值

表 35 极限值

遵循绝对最大额定系统规范 (IEC 60134)

符号	参数	条件	最小	最大	单位
V_{DD}	电源电压		- 0.3	+ 4.6	V
V_I	电压范围 (任何输入)		- 0.3	+ 5.5 ^[1]	V
I_I	DC 输入电流 (任何输入)		- 10	+ 10	mA
I_O	DC 输出电流 (任何输出)		- 10	+ 10	mA
P_{tot}	总功耗		-	300	mW
P_O	每个输出的功耗		-	50	mW
T_{amb}	工作温度		- 40	+ 85	°C
T_{stg}	存储温度		- 65	+ 150	°C

[1] 仅当电源电压存在时, 输入和输出上容限的 5.5V 稳态电压才有效。当电源电压不存在时, 输入和输出上容限的稳态电压为 4.6V。

13. 静态特性

表 36 静态特性

$V_{DD} = (2.5V \pm 0.2V)$ 或 $(3.3V \pm 0.3V)$; $T_{amb} = -40^{\circ}C \sim +85^{\circ}C$; 除非特别说明。

符号	参数	条件	V _{DD} =2.5V		V _{DD} =3.3V		单位
			最小	最大	最小	最大	
电源电压							
V _{DD}	供电电压		2.3	2.7	3.0	3.6	V
I _{DD}	供电电流	操作；无负载	-	6.0	-	6.0	mA
输入 I2C/ SPI, RX, CTS							
V _{IH}	高电平输入电压		1.6	5.5 ^[1]	2.0	5.5 ^[1]	V
V _{IL}	低电平输入电压		-	0.6	-	0.8	V
I _L	漏电流	输入；V _I =0V或 5.5V ^[1]	-	1	-	1	μA
C _i	输入电容		-	3	-	3	pF
输出 TX, RTS, SO							
V _{OH}	高电平输出电压	I _{OH} = - 400μA	1.85	-	-	-	V
		I _{OH} = - 4mA	-	-	2.4	-	V
V _{OL}	低电平输出电压	I _{OL} = 1.6mA	-	0.4	-	-	V
		I _{OL} = 4mA	-	-	-	0.4	V
C _O	输出电容		-	4	-	4	pF

续上表

符号	参数	条件	V _{DD} =2.5V		V _{DD} =3.3V		单位
			最小	最大	最小	最大	
输入/输出 GPIO0~GPIO7							
V _{IH}	高电平输入电压		1.6	5.5 ^[1]	2.0	5.5 ^[1]	V
V _{IL}	低电平输入电压		-	0.6	-	0.8	V
V _{OH}	高电平输出电压	I _{OH} = - 400μA	1.85	-	-	-	V
		I _{OH} = - 4mA	-	-	2.4	-	V
V _{OL}	低电平输出电压	I _{OL} = 1.6mA	-	0.4	-	-	V
		I _{OL} = 4mA	-	-	-	0.4	V
I _L	漏电流	输入； V _I =0V或 5.5V ^[1]	-	1	-	1	μA
C _O	输出电容		-	4	-	4	pF
输出 $\overline{\text{IRQ}}$							
V _{OL}	低电平输出电压	I _{OL} = 1.6mA	-	0.4	-	-	V
		I _{OL} = 4mA	-	-	-	0.4	V
C _O	输出电容		-	4	-	4	pF
I ² C总线输入/输出SDA							
V _{IH}	高电平输入电压		1.6	5.5 ^[1]	2.0	5.5 ^[1]	V
V _{IL}	低电平输入电压		-	0.6	-	0.8	V
V _{OL}	低电平输出电压	I _{OL} = 1.6mA	-	0.4	-	-	V
		I _{OL} = 4mA	-	-	-	0.4	V
I _L	漏电流	输入； V _I =0V或 5.5V ^[1]	-	10	-	10	μA
C _O	输出电容		-	7	-	7	pF
I ² C总线输入SCL, $\overline{\text{CS}}$ /A0, SI/A1							
V _{IH}	高电平输入电压		1.6	5.5 ^[1]	2.0	5.5 ^[1]	V
V _{IL}	低电平输入电压		-	0.6	-	0.8	V
I _L	漏电流	输入； V _I =0V或 5.5V ^[1]	-	10	-	10	μA
C _i	输入电容		-	7	-	7	pF
时钟输入XTAL1 ^[2]							
V _{IH}	高电平输入电压		1.8	5.5 ^[1]	2.4	5.5 ^[1]	V
V _{IL}	低电平输入电压		-	0.45	-	0.6	V
I _L	漏电流	输入； V _I =0V或 5.5V ^[1]	- 30	+30	- 30	+30	μA
C _i	输入电容		-	3	-	3	pF
睡眠电流							
I _{DD(sleep)}	睡眠电流	输入为V _{DD} 或地	-	30	-	30	μA

[1] 仅当电源电压存在时, 输入和输出上容限的 5.5V 稳态电压才有效。当电源电压不存在时, 输入和输出上容限的稳态电压为 3.8V。

[2] 当 XTAL1 由外部时钟驱动时, XTAL2 应悬空。

14. 动态特性

表 37 I²C 总线时序规范

所有时序极限值在电源电压, 操作温度和输出负载范围内都有效; $V_{DD} = (2.5V \pm 0.2V)$ 或 $(3.3V \pm 0.3V)$; $T_{amb} = -40^{\circ}C \sim +85^{\circ}C$; 参考 $V_{SS} \sim V_{DD}$ 输入电压下的 V_{IL} 和 V_{IH} 。所有输出负载 = 25pF, 而 SDA 输出负载 = 400pF 除外。

符号	参数	条件	标准模式 I ² C 总线		高速模式 I ² C 总线		单位
			最小	最大	最小	最大	
f_{SCL}	操作频率	[1]	0	100	0	400	kHz
t_{BUF}	停止和起始之间的总线空闲时间		4.7	-	1.3	-	μs
$t_{HD:STA}$	起始条件保持时间		4.0	-	0.6	-	μs
$t_{SU:STA}$	起始条件设置时间		4.7	-	0.6	-	μs
$t_{SU:STO}$	停止条件设置时间		4.7	-	0.6	-	μs
$t_{HD:DAT}$	数据保持时间		0	-	0	-	ns
$t_{VD:ACK}$	数据有效应答		-	0.6	-	0.6	μs
$t_{VD:DAT}$	SCL 低到数据输出有效		-	0.6	-	0.6	ns
$t_{SU:DAT}$	数据设置时间		250	-	150	-	ns
t_{LOW}	时钟低电平周期		4.7	-	1.3	-	μs
t_{HIGH}	时钟高电平周期		4.0	-	0.6	-	μs
t_f	时钟/数据下降时间		-	300	-	300	ns
t_r	时钟/数据上升时间		-	1000	-	300	ns
t_{sp}	干扰脉宽容限		-	50	-	50	ns
t_{d1}	I ² C 总线 GPIO 输出有效		0.5	-	0.5	-	μs
t_{d2}	I ² C 总线 modem 输入中断有效		0.2	-	0.2	-	μs
t_{d3}	I ² C 总线 modem 输入中断清除		0.2	-	0.2	-	μs
t_{d4}	I2C 输入管脚中断有效		0.2	-	0.2	-	μs
t_{d5}	I2C 输入管脚中断清除		0.2	-	0.2	-	μs
t_{d6}	I ² C 总线接收中断有效		0.2	-	0.2	-	μs
t_{d7}	I ² C 总线接收中断清除		0.2	-	0.2	-	μs
t_{d8}	I ² C 总线发送中断清除		1.0	-	0.5	-	μs
t_{d15}	复位后 SCL 延时	[3]	3	-	3	-	μs

[1] SCL 时钟频率的最小值由总线超时特性限制, 若 SDA 保持低电平持续最少 25ms, 那么该特性复位串行总线接口。

[2] I²C 总线规范的详细描述 (含有应用) 在小册子 “I²C 总线以及如何使用” 中给出。该小册子可使用代码 9398 393 40011 来订购。

[3] 2×1 个时钟或 3 μs , 无论哪个时间更短都可以。

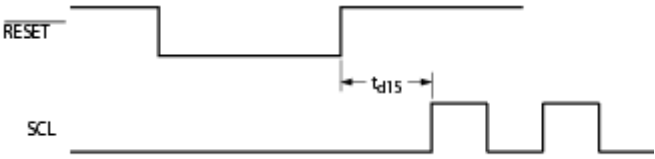
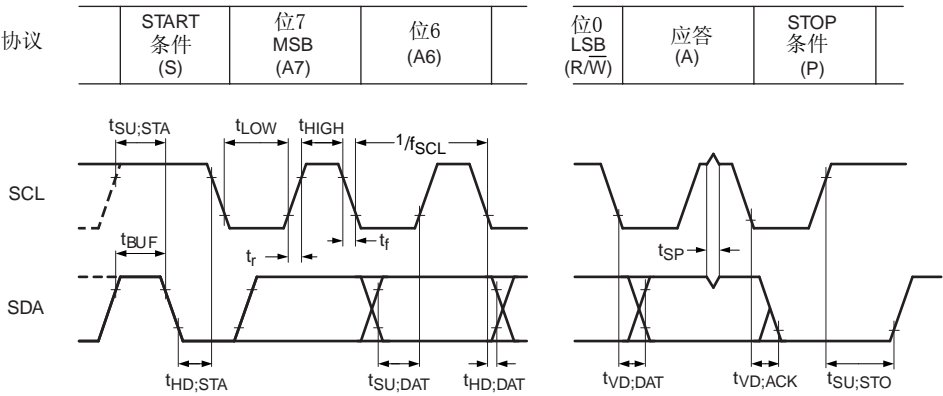


图 21 复位后 SCL 延时



上升和下降时间指 V_{IL} 和 V_{IH} 。

图 22 I²C总线时序框图

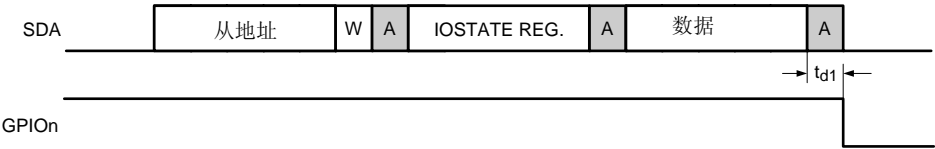


图 23 写到输出

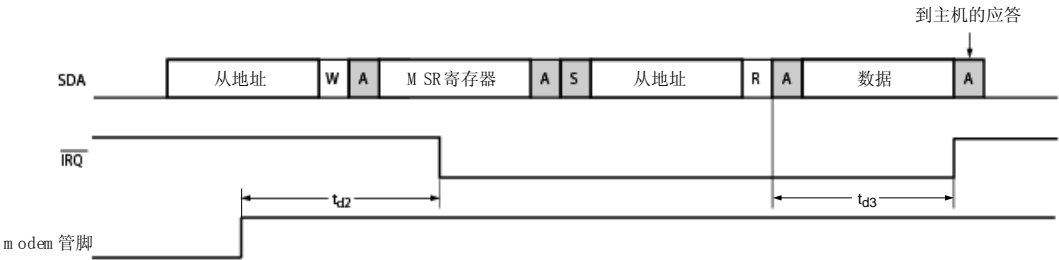


图 24 modem 输入管脚中断

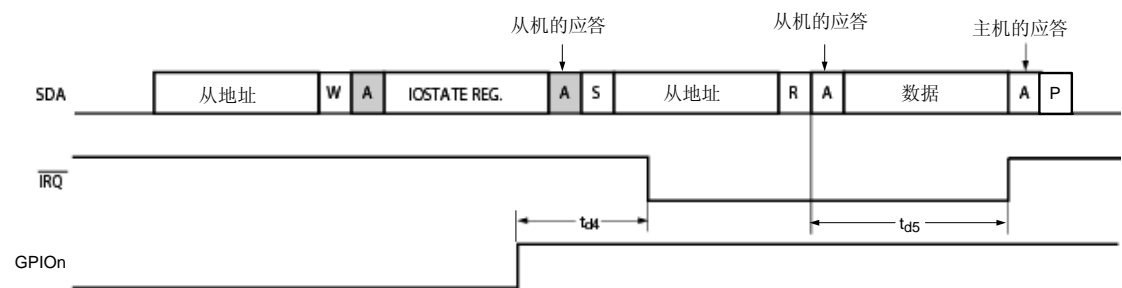


图 25 GPIO 管脚中断

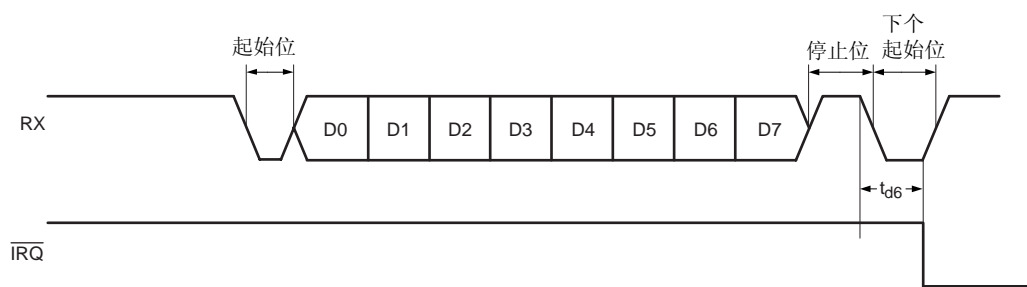


图 26 接收中断

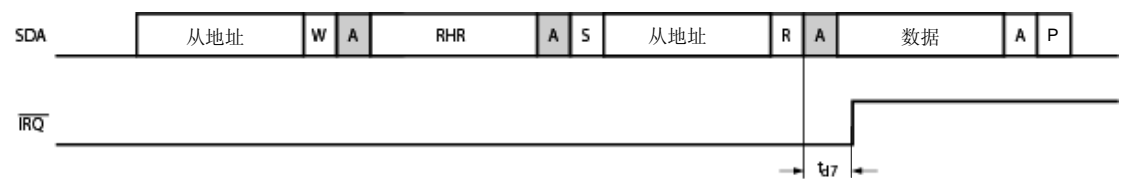


图 27 接收中断清除



图 28 发送中断清除

表 38 f_{XTAL} 动态特性

$V_{DD} = (2.5V \pm 0.2V)$ 或 $(3.3V \pm 0.3V)$; $T_{amb} = -40^{\circ}C \sim +85^{\circ}C$;

符号	参数	条件	$V_{DD}=2.5V$		$V_{DD}=3.3V$		单位
			最小	最大	最小	最大	
t_{w1}, t_{w2}	时钟脉冲持续时间		10	-	6	-	ns
f_{XTAL}	振荡器/时钟频率	[1] [2]	-	48	-	80	MHz

[1] 应用于外部时钟，晶体振荡器最大值为 24MHz。

$$[2] f_{XTAL} = \frac{1}{t_{3w}}$$

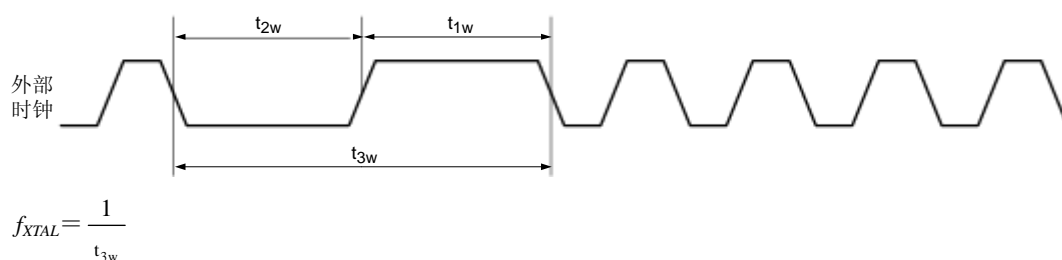


图 29 外部时钟时序

表 39 SPI 总线时序规范

所有时序极限值在电源电压, 操作温度和输出负载范围内都有效; $V_{DD} = (2.5V \pm 0.2V)$ 或 $(3.3V \pm 0.3V)$; $T_{amb} = -40^{\circ}C \sim +85^{\circ}C$; 参考 $V_{SS} \sim V_{DD}$ 输入电压下的 V_{IL} 和 V_{IH} 。所有输出负载 = 25pF, 除非特别说明。

符号	参数	条件	最小	典型	最大	单位
t_{TR}	\overline{CS} 高到 SO 三态	$C_L = 100pF$	-	-	100	ns
t_{CSS}	\overline{CS} 到 SCLK 设置时间		100	-	-	ns
t_{CSH}	\overline{CS} 到 SCLK 保持时间		20	-	-	ns
t_{DO}	SCLK 下降到 SO 有效	$C_L = 100pF$	-	-	100	ns
t_{DS}	SI 到 SCLK 设置时间		100	-	-	ns
t_{DH}	SI 到 SCLK 保持时间		20	-	-	ns
t_{CP}	SCLK 周期	$t_{CL} + t_{CH}$	250	-	-	ns
t_{CH}	SCLK 高电平时间		100	-	-	ns
t_{CL}	SCLK 低电平时间		100	-	-	ns
t_{CSW}	\overline{CS} 高脉冲宽度		200	-	-	ns
t_{d9}	SPI 输出数据有效		200	-	-	ns
t_{d10}	SPI modem 输出数据有效		200	-	-	ns
t_{d11}	SPI 发送中断清除		200	-	-	ns
t_{d12}	SPI modem 输入中断清除		200	-	-	ns
t_{d13}	\overline{SPI} 输入管脚中断清除		200	-	-	ns
t_{d14}	SPI 接收中断清除		200	-	-	ns

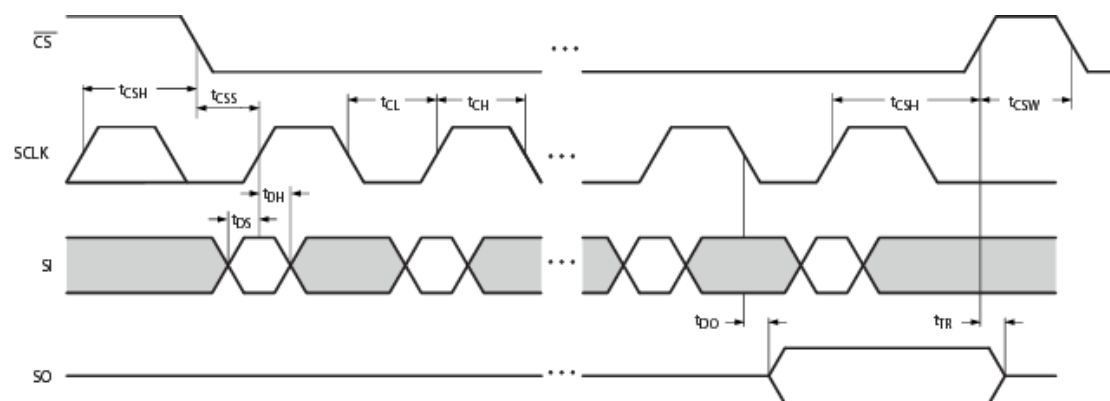
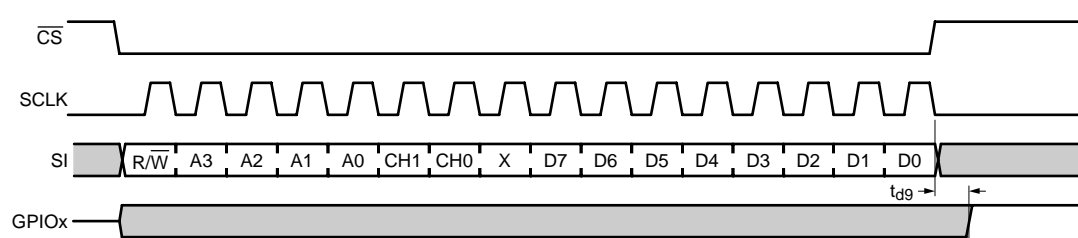
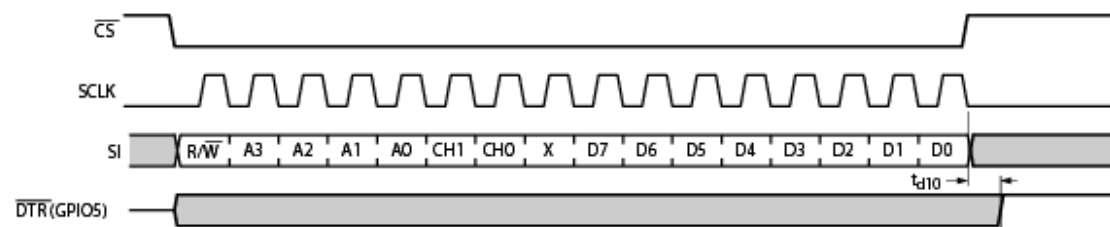


图 30 详细的 SPI 总线时序



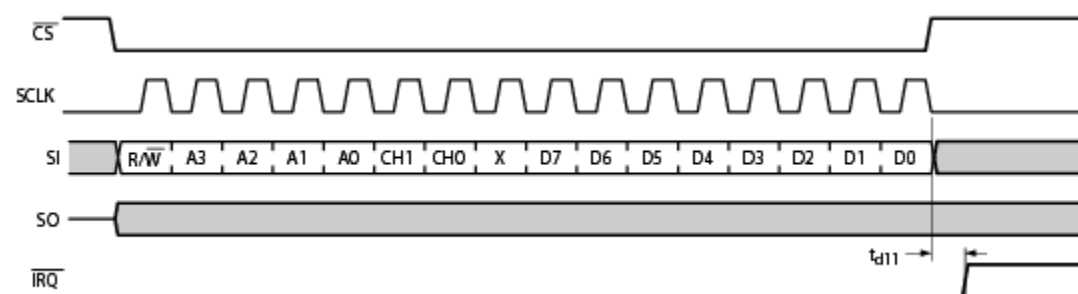
$R/\overline{W} = 0$; $A[3:0] = \text{IOState}(0x0B)$; $CH[1:0] = 00$ 用于通道 A; $CH[1:0] = 01$ 用于通道 B

图 31 SPI 写 IOState 到 GPIO 切换



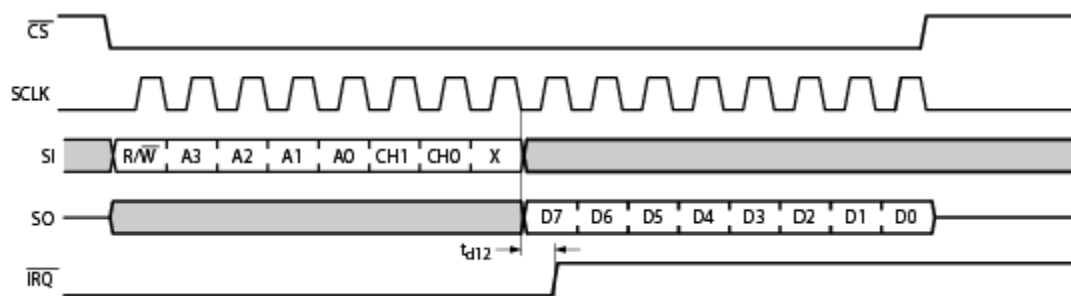
$R/\overline{W} = 0$; $A[3:0] = \text{MCR}(0x04)$; $CH[1:0] = 00$ 用于通道 A; $CH[1:0] = 01$ 用于通道 B

图 32 SPI 写 MCR 到 DTR 输出切换



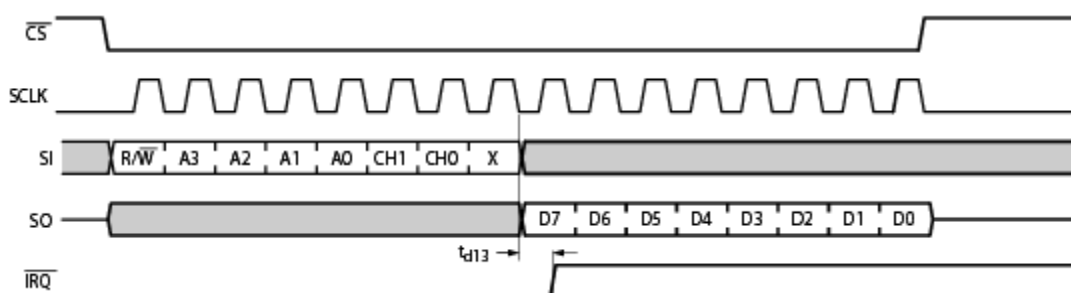
$R/\overline{W} = 0$; $A[3:0] = \text{THR}(0x00)$; $CH[1:0] = 00$ 用于通道 A; $CH[1:0] = 01$ 用于通道 B

图 33 SPI 写 THR 到清除 TX INT



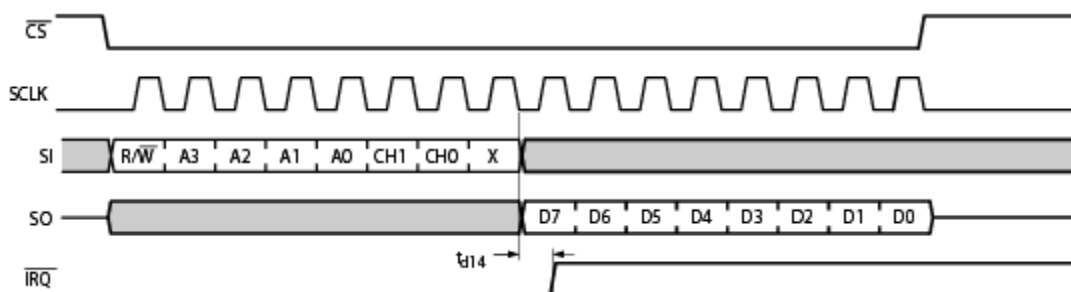
$R/\overline{W} = 1$; $A[3:0] = \text{MSR}(0x06)$; $\text{CH}[1:0] = 00$ 用于通道 A; $\text{CH}[1:0] = 01$ 用于通道 B

图 34 读 MSR 到清除 modem INT



$R/\overline{W} = 1$; $A[3:0] = \text{IOState}(0x0B)$; $\text{CH}[1:0] = 00$ 用于通道 A; $\text{CH}[1:0] = 01$ 用于通道 B

图 35 读 IOState 到清除 GPIO INT

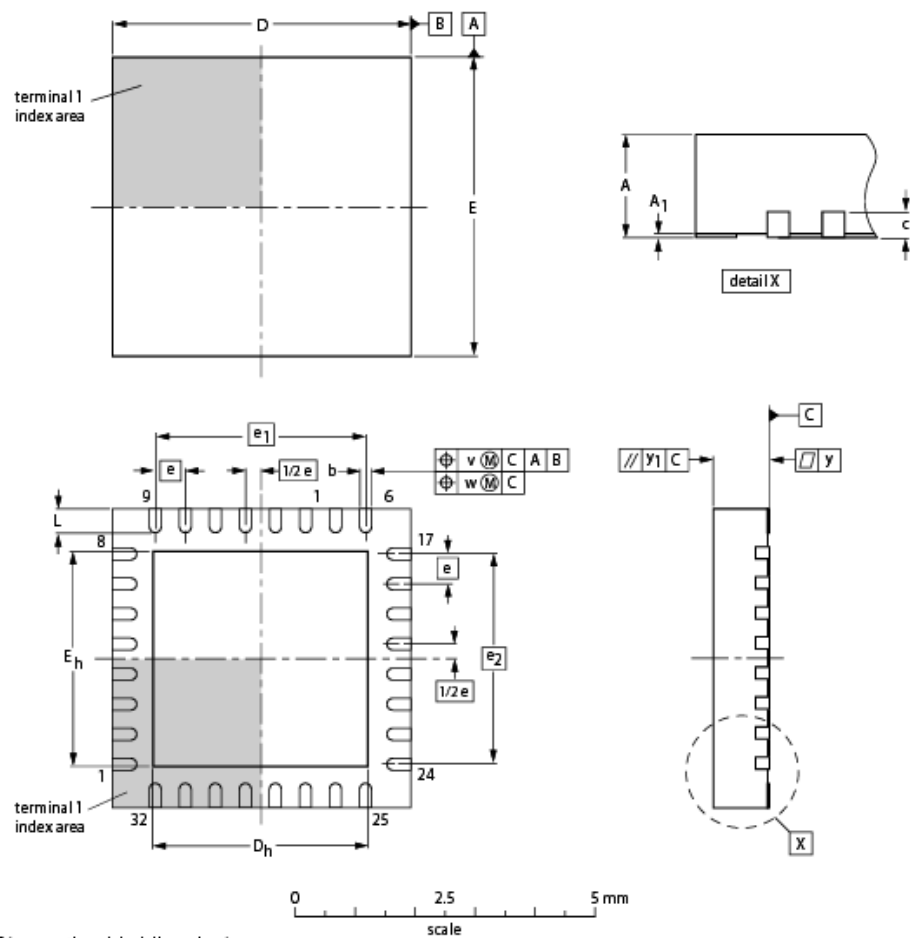


$R/\overline{W} = 1$; $A[3:0] = \text{RHR}(0x00)$; $\text{CH}[1:0] = 00$ 用于通道 A; $\text{CH}[1:0] = 01$ 用于通道 B

图 36 读 RHR 到清除 RX INT

15.表面封装

HVQFN32: 无引脚; 32 端; 本体 5×5×0.85mm



DIMENSIONS (mm are the original dimensions)

UNIT	A ⁽¹⁾ max.	A ₁	b	c	D ⁽¹⁾	D _h	E ⁽¹⁾	E _h	e	e ₁	e ₂	L	v	w	y	y ₁
mm	1	0.05 0.00	0.30 0.18	0.2	5.1 4.9	3.75 3.45	5.1 4.9	3.75 3.45	0.5	3.5	3.5	0.5 0.3	0.1	0.05	0.05	0.1

Note

1. Plastic or metal protrusions of 0.075 mm maximum per side are not included.


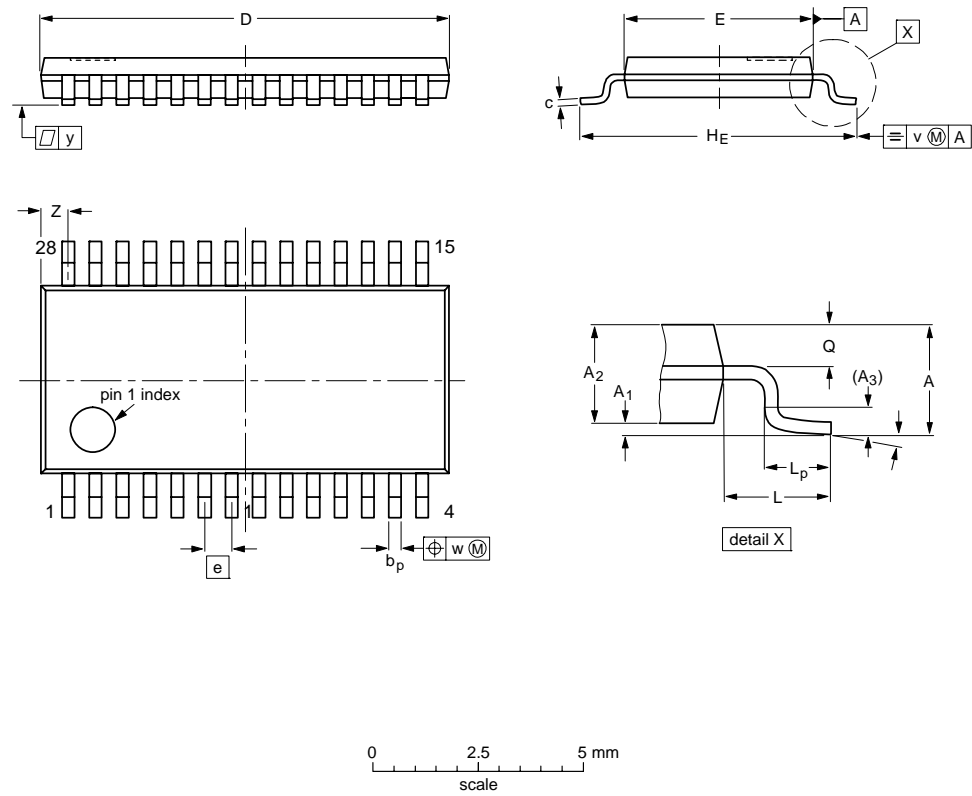
OUTLINE VERSION	REFERENCES				EUROPEAN PROJECTION	ISSUE DATE
	IEC	JEDEC	JEITA			
SOT617-3	---	MO-220	---			-02-04-18- 02-10-22

图 37 表面封装 SOT617-3 (HVQFN32)

TSSOP28: 28 脚; 本体宽度 4.4mm



DIMENSIONS (mm are the original dimensions)

UNIT	A max.	A ₁	A ₂	A ₃	b _p	c	D ⁽¹⁾	E ⁽²⁾	e	H _E	L	L _p	Q	v	w	γZ	(1)	
mm	1.1	0.15 0.05	0.95 0.80	0.25	0.30 0.19	0.2 0.1	9.8 9.6	4.5 4.3	0.65	6.6 6.2	1	0.75 0.50	0.4 0.3	0.2	0.13	0.1	0.8 0.5	8° 0°

Notes

1. Plastic or metal protrusions of 0.15 mm maximum per side are not included.
2. Plastic interlead protrusions of 0.25 mm maximum per side are not included.


OUTLINE VERSION	REFERENCES				EUROPEAN PROJECTION	ISSUE DATE
	IEC	JEDEC	JEITA			
SOT361-1		MO-153				-99-12-27- 03-02-19

图 38 表面封装 SOT361-1 (TSSOP28)